```
package grapes_treat;
/*
 *Control
 */
import java_kqml.*;
import java.awt.*;
import java.applet.*;
import class.om.*;
import java.util.StringTokenizer;
import java.net.*;
import java.io.*;
public class Control extends Applet{
//Interface elements
Button runButton = new Button("Click here to start treatment ES");
ESClient esC ;
//Parameter declarations
        private String hostName;
        private int portNo;
        private final String PARAM_host = "thost";
        private final String PARAM_port = "tport";
//HC Declaration:
        HCNodes ex = new HCNodes();
public String getAppletInfo()
        {return "Name: grapes treatment ES\r\n" +   "" ;     }
public String[][] getParameterInfo()
        {String[][] info =
                {
                        { PARAM_host, "String", "" },
                        { PARAM_port, "int", "" },};
                return info;}

public void init(){
        String param;
        param = getParameter(PARAM_host);
        if (param != null)
                hostName = param;
                param = getParameter(PARAM_port);
        if (param != null)
        portNo = Integer.parseInt(param);
        setFont(new Font("Helvetica", Font.PLAIN, 14));
        runButton.setSize(350,350);
        add(runButton);}

public void run(String []disconfL,String diagoutSt,String GStage) {
        String ds="disorder-value-";
        int dlen=ds.length();
        String digoutp=diagoutSt.substring(dlen);
        ex.HCCkb.GDB.reset();
        System.out.println("=============================================================");
        System.out.println("STARTED EXECUTION OF COORDINATION AGENT AT TREATMENT SESSION");
        runButton.disable();
        ex.HCCkb.value.setVal(digoutp);
        ex.HCCkb.value_val=disconfL;
        ex.HCCkb.disorder_exist.setVal("yes");
        ex.HCCkb.growth_stage.setVal(GStage);
        ex.run(); //run HC
        String out = ex.HCCkb.GDB.getObjAll();
        String diag;
        //String out="'(tell :language prolog :content [plantation-irrigation_system-dripping,vine_voo-
growth_stage-flowering,disorder-value-[wilt_and_root_rots],wilt_and_root_rots-confirmed
confirmed,disorder-value-[nitrogen_def],nitrogen_def-confirmed-confirmed,null-disorder_exist-yes]. )'";
        if(out.equals("[null-disorder_exist-no]"));
else
{
        ESClient esC =  new ESClient(hostName,portNo);
        // form the sending message
    out = "'(tell :language prolog :content "+out+". )'" ;
        esC.out(out);
        String in =esC.in();
        esC.out("bye");
        //System.out.println("OUTPUT FROM COORDINATION AGENT TO TREATMENT AGENT:");
        System.out.println("HOST NAME:="+hostName);
        System.out.println("PORT NUMBER="+portNo);
        System.out.println("COORDINATION OUTPUT="+out);
        System.out.println("OUTPUT FROM TREATMENT AGENT:");
        System.out.println("OUTPUT:"+in);
        KQMLmessage message = new KQMLmessage(in);
        String performative = message.getValue("performative");
        String content = message.getValue("content");
        //System.out.println("Performative="+performative);
        //System.out.println("Content="+content);
        testInput(performative,content);
esC.CloseClient();}
        runButton.enable();}
```

```
public boolean action(Event evt, Object arg) {
        if (evt.target == runButton){
                showStatus("wait...");
        //      run();
                }/* else if(evt.target == resetButton)
                        ex.McCkb.GD3.reset(); */
      return false; }
  public void start() {};
  public void destroy() {
       repaint();       }
  public void testinput(String performative, String content){
        if(performative.equals("reply")){
          actionReply(content);
          performative=null;
          content=null;   }
        else if(performative.equals("Sorry1")){
        showStatus("Sorry I have no treatment for this disorder...");
        actionSorry1(content);
         //System.out.println("end action sorry1"); }
        else if(performative.equals("Sorry2")){
        showStatus("Please consult diagnosis...");  }        }

public String[] parseStr(String inStr, String delm) {
                StringTokenizer st = new StringTokenizer(inStr,delm,false);
        String[] newList = new String[st.countTokens()];
                int  count = 0;
                while (st.hasMoreTokens()) {
                        newList[count] =st.nextToken();
                        if (newList[count].equals("[]"))
                                newList[count]="";
                //System.out.println("the output content is "+newList[count] );
                        count ++;  }
                return newList;   }
        public void actionReply(String content){
        //System.out.println("begin action treatment reply");
        String field1 = content.substring(1); /* remove : */
        int k=field1.lastIndexOf(")");
        String field2=field1.substring(0,k);
        String[] contentparse1 = parseStr(field2,",");
        String [] disorderL=new String[contentparse1.length];
        int j=0;
        for (int i=0;i< contentparse1.length;i++){
        String[] contentparse2 = parseStr(contentparse1[i],"--");
        disorderL[j]=contentparse2[0];
        j++;}
        ListPanel LP= new ListPanel(disorderL,contentparse1);
        LP.show();
        //      System.out.println("end action treatment reply");
        // runButton.enable();     }
        public void actionSorry1(String content){
        showStatus("SORRY I HAVE NO treatment for those disorder...");
        String sorry = "Sorry I have no treatment for this disorder...";
        Err errm =new Err("Treatment results", sorry,new Frame());
        errm.setFont(new Font("Helvetica", Font.PLAIN, 12));
        errm.setBackground(Color.white);
        errm.setForeground(Color.blue);
        errm.setLocation(200,200);
        errm.resize(450,100);
        errm.show();} }
/* ------------------------------------ */

package grapes_treat;
// treatment communication model
import class.cm.*;

class Node0 extends Node {
     hcgdb ckb;
     public Node0(Node p, hcgdb ckb) {
          super( p);
          this.ckb = ckb;      }
     public int eval() {
          if( ckb.disorder_exist.getValue())
               return 5;
          return 0;      }}

class Node1 extends Node {
     hcgdb ckb;
     public Node1(Node p, hcgdb ckb) {
          super( p);
          this.ckb = ckb;      }

     public int eval() {
                if ( (ckb.value.MemberChk("root_knot_nematode")||
                        ckb.value.MemberChk("root_lesion_nematode")||
```

```
                    ckb.value.MemberChk("reniform_nematode")||
                    ckb.value.MemberChk("dagger_nematode"))      &&
                  ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                    ((ckb.infection.getValue()).compareTo("medium")== 0)&&
                    ( ((ckb.nema_material.getValue()).compareTo("vidate_ec_24")== 0)||
                      ((ckb.nema_material.getValue()).compareTo("mokap_granules_10")== 0)||
                ((ckb.nema_material.getValue()).compareTo("furadan_granules_10")== 0) ))
                    return 6;
                return 0;     }}

class Node2 extends Node {
     hegdb ckb;
     public Node2(Node p, hegdb ckb) {
          super( p);
          this.ckb = ckb;     }
     public int eval() {
               if ( ((ckb.nema_material.getValue()).compareTo("vidate_ec_24")== 0)&&
                       (((ckb.irrigation_system.getValue()).compareTo("flooding")== 0)||
                       ((ckb.irrigation_system.getValue()).compareTo("dripping")== 0) ))
                       return 6;
               return 0;     }}

class Node7 extends Node {
     hegdb ckb;
     public Node7(Node p, hegdb ckb) {
          super( p);
          this.ckb = ckb;     }
     public int eval() {
               if ( (ckb.value.MemberChk("dagger_nematode")||
                       ckb.value.MemberChk("root_lesion_nematode")||
                       ckb.value.MemberChk("reniform_nematode")&&
                      ((ckb.value.getValue()).compareTo("root_knot_nematode")!=0)&&
                      ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                       ((ckb.infection.getValue()).compareTo("high")== 0)&&
                      ( ((ckb.nema_material.getValue()).compareTo("vidate_ec_24")== 0)||
                        ((ckb.nema_material.getValue()).compareTo("mokap_granules_10")== 0)||
                        ((ckb.nema_material.getValue()).compareTo("furadan_granules_10")== 0)))
                    return 6;
               return 0;     }}

class Node22 extends Node {
     hegdb ckb;
     public Node22(Node p, hegdb ckb) {
          super( p);
          this.ckb = ckb;     }
     public int eval() {
               if ( ((ckb.nema_material.getValue()).compareTo("vidate_ec_24")== 0)&&
                       (((ckb.irrigation_system.getValue()).compareTo("flooding")== 0)||
                       ((ckb.irrigation_system.getValue()).compareTo("dripping")== 0)))
                    return 6;
                    return 0;     }}

class Node3 extends Node {
     hegdb ckb;
     public Node3(Node p, hegdb ckb) {
          super( p);
          this.ckb = ckb;     }
     public int eval() {
               if ( ckb.value.MemberChk("powdery_mildew")&&
                   ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                        ( ((ckb.powd_material.getValue()).compareTo("kemazy")== 0)||
                          ((ckb.powd_material.getValue()).compareTo("topsin_m10")== 0)||
                          ((ckb.powd_material.getValue()).compareTo("dworado")== 0)||
                          ((ckb.powd_material.getValue()).compareTo("sumi8")== 0)))
                        return 6;
               return 0;     }}

class Node4 extends Node {
     hegdb ckb;
     public Node4(Node p, hegdb ckb) {
          super( p);
          this.ckb = ckb;     }
     public int eval() {
               if ( ckb.value.MemberChk("downy_mildew")&&
                   ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                        ( ((ckb.down_material.getValue()).compareTo("antracol")== 0)||
                          ((ckb.down_material.getValue()).compareTo("radomil")== 0)))
                      return 6;
                        return 0;     }}

class Node5 extends Node {
     hegdb ckb;
     public Node5(Node p, hegdb ckb) {
          super( p);
```

```
            this.ckb = ckb;     }
        public int eval() {
                    if ( (ckb.value.MemberChk("vine_thrips")||
                            ckb.value.MemberChk("vine_grape_moth") ||
                              ckb.value.MemberChk("honeydew_moth") &&
                              !(ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                            | !(ckb.vth_material.getValue()).compareTo("anthio")== 0)||
                              !(ckb.vth_material.getValue()).compareTo("acmethoon")== 0)))
                             return 6;
                            return 0;     ||
    class Node6 extends Node {
         hcgdb ckb;
         public Node6(Node p, hcgdb ckb) {
              super( p);
              this.ckb = ckb;     }
         public int eval() {
                    if ( ckb.value.MemberChk("zinc_def")&&
                        ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                             ( ((ckb.zinc_material.getValue()).compareTo("zinc sulphat")== 0)||
                               ((ckb.zinc_material.getValue()).compareTo("zinc chelate")== 0)))
                                 return 6;
                             return 0;})

         class Node8 extends Node {
         hcgdb ckb;
         public Node8(Node p, hcgdb ckb) {
              super( p);
              this.ckb = ckb;     }
         public int eval() {
                    if ( (ckb.value.MemberChk("nitrogen_def")||
                              ckb.value.MemberChk("potassium_def"))&&
                        ((ckb.growth_stage.getValue()).compareTo("dormance")!= 0)&&
                             (((ckb.irrigation_system.getValue()).compareTo("flooding")== 0)||
                               ((ckb.irrigation_system.getValue()).compareTo("dripping")== 0)))
                             return 5;
                    return 0;})
    public class HCNodes extends HC {
         hcgdb HcCkb = new hcgdb();
         Node0 n0 = new Node0(node00, HcCkb);
         Node1 n1 = new Node1(n0, HcCkb);
         Node2 n2 = new Node2(n1, HcCkb);
         Node7 n7 = new Node7(n0, HcCkb);
         Node22 n22 = new Node22(n7, HcCkb);
        Node3 n3 = new Node3(n0, HcCkb);
         Node4 n4 = new Node4(n0, HcCkb);
         Node5 n5 = new Node5(n0, HcCkb);
         Node6 n6 = new Node6(n0, HcCkb);
        Node8 n8 = new Node8(n0, HcCkb);
    }
    /*******************/
    package grapes_treat;
    // This code is generated automatically using XBGL tool for HC
    import clees.cn.*;
    public class hcgdb extends CKB {
         YesNoVar disorder_exist;
              OneOfVar growth_stage;
              OneOfVar nema_material;
         OneOfVar powd_material;
              OneOfVar down_material;
              OneOfVar vth_material;
              OneOfVar zinc_material;
         OneOfVar irrigation_system;
              OneOfVar infection;
              ManyOfVar value;
         public hcgdb() {
             disorder_exist = new YesNoVar("disorder_exist");
              disorder_exist.setQuestion("Do you have a problem ");
               CDB.add(disorder_exist);
             String[] arg0= {"vidate_ec_20","mokap_granules_10","firadan_granules_10"};
             nema_material = new OneOfVar("nema_material", arg0 );
              nema_material.setQuestion("What is the nematode used material");
              NEMA_material.setObjname("treatment_op");
              CDB.add(nema_material);
               String[] arg1 = {"kemany","topsin_m70","dwarde","sumi8"};
              powd_material = new OneOfVar("powd_material", arg1 );
              powd_material.setQuestion("What is the powdry_mildew used material");
              powd_material.setObjname("treatment_op");
              CDB.add(powd_material);
                        String[] arg2 = {"antracoll","redomel"};
              down_material = new OneOfVar("down_material", arg2 );
              down_material.setQuestion("What is the downy_mildew used material");
              down_material.setObjname("treatment_op");
              CDB.add(down_material);
                       String[] arg3 = {"anthio","acmethoon"};
```

```
            vth_material = new OneOfVar("vth_material", arg3 );
            vth_material.setQuestion("What is the used material");
            vth_material.setObjname("treatment_op");
            GDB.add(vth_material);
             String[] arg4 = {"zinc_sulphat","zinc_choleca"};
            zinc_material = new OneOfVar("zinc_material", arg4 );
            zinc_material.setQuestion("What is the zinc used material");
            zinc_material.setObjname("treatment_op");
            GDB.add(zinc_material);
            String[] arg5 = {"flooding","dripping"};
           irrigation_system = new OneOfVar("irrigation_system", arg5 );
            irrigation_system.setQuestion("What is the irrigation_system");
            irrigation_system.setObjname("plantation");
            GDB.add(irrigation_system);
             String[] arg6 = {"dormance", "fruiting", "flowering", "vegetative","harvest","post_harvest"};
            growth_stage = new OneOfVar("growth_stage", arg6 );
            growth_stage.setQuestion("What is the growth stage of vine");
            growth_stage.setObjname("vine_obs");
                 GDB.add(growth_stage);
             String[] arg7 = {"heigh","meduim","low"};
            infection = new OneOfVar("infection", arg7 );
            infection.setQuestion["What is the infection"];
            infection.setObjname("disorder");
                 GDB.add(infection);
             String[] arg8 = {"nitrogen_def","potassium_def","zinc_def","dagger_nematode",
"root_lesion_nematode", "vine_grape_moth", "vine_thrips", "fruit_rots", "downy_mildew", "powdry_mildew",
"wilt_and_root_rots", "honeydew_moth", "scale", "beely_bugs" , "reniform_nematode", "root_knot_nematode",
"eriophyes_vitis", "mite", "land_snails", "vine_borers", "peach_borers"};
            value= new ManyOfVar("value", arg8 );
            value.setQuestion("what is the disorder");
                    value.setObjname("disorder");
            GDB.add(value);    ]}

/*****************************/
package grapet_diag;
/*
  * presentation component
 * AskTreatpanel
 */
import java.awt.*;
import java.applet.*;
import class.cm.*;
import java.util.StringTokenizer;

public class AskTreatPanel extends Dialog
{
        protected Label TitleL = new Label("The Confirmed Disorders:");
        protected Label DisorderL = new Label("Disorder:");
        protected Label ConfirmedL = new Label("Confirmed");
        protected Label AskL = new Label("Do you want to run treatment");
        protected Button ok = new Button("Yes");
        protected Button no = new Button("Cancel");
        protected Label cont = new Label();
        String diagOut;
        boolean result;
public AskTreatPanel(String[] DiagOut) {
        super((new Frame()), "Result of Diagnosis session", true);
        Label[] labels1 = new Label[12];
        setFont(new Font("Helvetica", Font.PLAIN, 14));
                setBackground(Color.cyan);
                setForeground(Color.blue);
                Panel p1= new Panel();
                p1.setLayout(new FlowLayout(FlowLayout.LEFT));
                TitleL.setAlignment(Label.LEFT);
                p1.add(TitleL);
                add("North", p1);
                Panel p2 = new Panel();
                p2.setLayout(new GridLayout(0,1,2,2));
                DisorderL.setBackground(Color.gray);
                ConfirmedL.setBackground(Color.gray);
                p2.add(DisorderL);
         for(int i=0;i<DiagOut.length;i++)     {
     labels1[i]=new Label();
        labels1[i].setText(DiagOut[i].toString ());
        p2.add(labels1[i]);       }
                add("Center", p2);
                Panel p3 = new Panel();
                p3.setLayout(new FlowLayout(FlowLayout.CENTER));
                p3.add(AskL);
                p3.add(ok);
                p3.add(no);
                add("South", p3);
        //initialize this dialog to its preferred size.
                resize(350,350);
```

```
                pack();  }
         public boolean action(Event event, Object arg) {
        if (event.target == ok){
                        result = true;
                        dispose();
                }else if (event.target == no){
                        result = false;
                        dispose();                }
                return false;      }}
package grapes_treat;
/*
 * presentation component
 * ListPanel
 *
 */
import java.awt.*;
import java.applet.*;
import java.util.StringTokenizer;
public class ListPanel extends Dialog
{       int length = 10;
        List  ls = new List(length,false);
        Button subButton = new Button("Quit");
        Label title=new Label("Double click on a disorder to see detail");
        String []CopyDisArryL;
        ListPanel(String [] DisorderL,String [|DisArryL) {
        super((new Frame()), "Disorder List", true);
        setFont(new Font("Helvetica", Font.PLAIN, 14));
        CopyDisArryL=DisArryL;
        Panel p1=new Panel();
        p1.setLayout(new FlowLayout(FlowLayout.LEFT));
        title.setBackground(Color.white);
        title.setForeground(Color.black);
        title.setAlignment(Label.LEFT);
                p1.add(title);
                add("North",p1);
                Panel p2 = new Panel();
                p2.setLayout(new FlowLayout(FlowLayout.CENTER));
                ls =copyL(ls,DisorderL);
                p2.add(ls);
                add("Center",p2);
                Panel p3 = new Panel();
                p3.setLayout(new GridLayout(1,0));
        //      p3.setLayout(new FlowLayout(FlowLayout.CENTER));
                p3.add(subButton);
                add("South",p3);
                setBackground(new Color(64,128,128));
                ls.setBackground(new Color(255,255,215));
                setForeground(Color.white);
                ls.setForeground(Color.darkGray);
                resize(350,350);
                pack();  }

        public void sendAndDisplay(String TreatList){
        //System.out.println("Begin send and display" );
        String[] DisTreatL = parseStr(TreatList,"=");
        newframe frame1;
        frame1=new newframe (DisTreatL);
        frame1.setBackground (Color.cyan);
        frame1.show(true);
        frame1.setResizable (false);
                //System.out.println("end send and display " );
        }

        public List copyL(List l, String[] strs)
        {       for(int i=0; i< strs.length; i++)
                l.addItem(strs[i]);
                return l;         }

                public String[] parseStr(String inStr,String delem) {
                StringTokenizer st = new StringTokenizer(inStr,delem,false);
                String[] newList = new String[st.countTokens()];
                int  count = 0;
                while (st.hasMoreTokens()) {
                        newList[count] =st.nextToken();
                        if (newList[count].equals("[]"))
                        newList[count]="";
                //System.out.println("the output content is "+newList[count] );
                        count ++;          }
                        return newList;   }

        public boolean action(Event evt, Object arg) {
        if (evt.target == subButton) {
                dispose();
                return true;
```

```
        }else
                if (evt.target instanceof List) {
                        int idx = la.getSelectedIndex();
                        if( idx > -1)
                        rendAndDisplay(CopyDisArryL[idx]);
                        else;
                                //showStatus("Please select a disorder first...");}
                return false;       }}

/**************************/
package grapes_treat;
/*
 * presentation component
 * newframe
 */

import java.awt.* ;
import java.awt.event.* ;
import java.util.StringTokenizer;
public class   newframe extends Frame
{

        protected Button Done = new Button("Quit");
        Button advice=new Button ("Advice");
        TextArea advice_area;
        public Panel title_panel,advice_panel,con_panel;
        public Label[] labels;
        public String [] adviceT;
        public newframe(String [DisTL) {
                        super("Result of Treament Session ");
                setfont(new Font("Helvetica", Font.PLAIN, 14));
                        adviceT=DisTL;
                        CreateUI();
                         add_comp(DisTL);
                 };
  public void add_comp(String[] str){
        str={"mite","material_nece","Vertamic","qty","251","unit","CM/100LM","method","","appl_time","",
"advice","hello"};
        Dimension framesize;
        int i = 0;
        setLayout(null);
        advice_area=new TextArea [];
        advice_area.setSize (400,150);
        Label    title = new Label();
        framesize=getSize ();
        title.setSize (framesize.width,50);
        title.setLocation (0,10);
title.setText ("                                The treatment result of disorder:"+str[1]);
        title.setBackground (Color.white );
        title.setForeground (Color.blue);
        add(title);
        add_label(str);
        advice.setLocation (10,250);
        advice.setSize (75,40);
        Done.setLocation (160,250);
        Done.setSize (50,50);
     Done.setLabel ("Quit");
         add(advice);
         add(Done);          }

  public void add_label(String[] strlabel){
  Label [] label1;
  int labellen=(strlabel.length)-3;
  label1 =new Label[labellen];
  TextField methodT=new TextField();
  int xstart;
  int ystart=50;
  for(int i=1;i<labellen;i++){
     label1[i]=new Label();
     if(i%2==0)
          xstart=140;
        else xstart=10;
    label1[i].setSize (150,20);
        if(xstart==10){
    if (i==1)
                label1[i].setLocation(xstart,ystart);
        else{
           label1[i].setLocation(xstart,i*30);
           //System.out.println("location"+xstart+(i*30));}}
        else{
                if (i==2)
                label1[i].setLocation(xstart,ystart);
                else{
```

```
                    labels[i].setLocation(xstart,(i-1)*30);
                    //System.out.println("location"+xstart+((i-1)*30));}}
              labels[i].setText(strlabel[i].toString ());
        add(labels[i]);
              }
     methodT.setLocation(160,210);
     methodT.setSize(180,30);
     methodT.setText(strlabel[0]);
     add(methodT);  }

     public String[] parseStr(String inStr, String delm) {
                    StringTokenizer st = new StringTokenizer(inStr,delm,false);
                    String[] newList = new String[st.countTokens()];
                    int  count = 0;
                    while (st.hasMoreTokens()) {
                            newList[count] =st.nextToken();
                            if (newList[count].equals("||"))
                                    newList[count]="";
//                  System.out.println("the output content is "+newList[count] ;                    count
++;         }
                            return newList;   }
    protected void CreateUI()
               {
                setSize(400,400);
                center();
                /*addWindowListener(new WindowAdapter()
                        {
                    public void WindowClosing(WindowEvent e)
                        {
                      if(e.COMPONENT_RESIZED)
                                                    dispose();
                        // System.exit (0);;} }

      public void center()    {
                  //Toolkit toolkit = //. Too_kIt();
                 Dimension ScreenSize;
                                    String str;
           ScreenSize=Toolkit.getDefaultToolkit().getScreenSize ();
                           Dimension  frameSize =getSize();
                int x = ScreenSize.width/2-frameSize.width/2;
                int y = ScreenSize.height/2-frameSize.height/2;
                           reshape (300,10,frameSize.width ,frameSize.height );

              }


public boolean action(Event e, Object arg) {
                    if (e.target==Done){
                              dispose();
                   return true;}
                   else if(e.target==advice){
          if(advice_area.isVisible ()){
            advice_area.setVisible (false);
            advice.setLabel("Advice");
            resize(400,400);
            advice_area.setText("");
            Done.setLocation (160,250);
              return true; }
          else
          {
             advice_area.setVisible (true);
             advice.setLabel("Hide Advice");
             resize(400,550);
        Done.setLocation (200,500);
             advice_area.setLocation (10,350);
            String[] adviceparser = parseStr(adviceT[10],"+");
            for(int i=0;i< adviceparser.length;i++)
                  advice_area.appendText(adviceparser[i]+"\n");
             add(advice_area);
             return true;
            }      }
          return false;     }}

/**********************/
package grapes_treat;
/*
 *presentation component
 * Err
 *
 */
import java.awt.*;
public class Err extends Dialog
{
        Label errText = new Label();
```

```
        Button ok = new Button("    OK    ");

    public Err(String title, String errString, Frame parent) {
                super(parent, title, true);
                Panel p1 = new Panel();
        p1.setLayout(new FlowLayout(FlowLayout.CENTER));
                errText.setText(errString);
                errText.setAlignment(Label.CENTER);
                p1.add(errText);
                 add("North", p1);
                Panel p2 = new Panel();
        p2.setLayout(new FlowLayout(FlowLayout.CENTER));
        p2.add(ok);
        add("South", p2);
                //Initialize this dialog to its preferred size.
                resize(200,100);
                pack();  }

         public boolean action(Event event, Object arg) {
                if (event.target == ok)
                 dispose();
                return true;    }  }


/*****************/
package java_kqml;
/*
*bilingual translator
* kqmlmessage
*/

import java.util.Hashtable;
import java.util.Enumeration;
import java.util.StringTokenizer;
import java.io.StreamTokenizer;
import java.io.InputStream;

public class KQMLmessage extends Language {
    protected Hashtable field_value_pairs;

  public KQMLmessage () {
    field_value_pairs = new Hashtable();  }

  public KQMLmessage(String message){
    this();
    parseString(message);  }

  public void parseString(String message){
        if(message.length() > 0){
      StringTokenizer st = new StringTokenizer(message,"\n\t\r\"\'()' #", true);
     st.nextToken();

     try{
        addFieldValuePair("performative",getNonDelimiter(st));
     } catch (Exception e){}
     while(st.hasMoreTokens()){
        String token = getNonDelimiter(st);
        if(token != null){
          String field = token.substring(1); /* remove : */
          String value = ParseExpression(st, 0, false, token);
          this.addFieldValuePair(field, value));    }  }  }

  protected String ParseExpression(StringTokenizer st, int num_paren, boolean string,
                        String last_token){
                    if(st.hasMoreTokens()){
          if(string){ /* we are inside of a string, must wait for double quote */
        String token = st.nextToken();
        if(token.equals("\"")){ /* end of string */
          if(num_paren == 0){ /* end of expression */
            return token;
          } else {    return( token.concat(ParseExpression(st,num_paren,false,token)));  }
        } else if(token.equals("\\")){ /* escape character */
          token = token.concat(st.nextToken());/* automatically take next token */
          return( token.concat(ParseExpression(st,num_paren,true,token)));
        } else {
          return(token.concat(ParseExpression(st,num_paren,true,token)));      }
            } else {
        String token = getNonWhiteSpace(st);
        if(token != null){
         if(token.equals("(")){ /* (<word> (<whitespace> <expression>}*) */
            if(num_paren == 0){/* get the <word>*/
              token = token.concat(st.nextToken()));      }
            num_paren++;
            return( token.concat(ParseExpression(st,num_paren,false,token)));
```

```
                    } else if(token.equals(")")){ /* close paren */
                  num_paren--;
                      if(num_paren == 0){
                        return token;
                      } else {
                  return( token.concat(ParseExpression(st,num_paren, false,token)));
                  } else if(token.equals("\'")){ /* '<expression> */
                    return( token.concat(ParseExpression(st,num_paren,false,token)));
                  } else if(token.equals("`")){ /* '<comma-expression> */
                    return( token.concat(ParseExpression(st,num_paren,false,token)));
                  } else if(token.equals("\"")){ /* "<stringchar>" */
                    return( token.concat(ParseExpression(st,num_paren,true,token)));
                  } else if(token.equals("#")){ /* #<digit><digits>"<cr>" */
                return(token.concat(LengthDelimitedString(st, num_paren)));
              } else { /* <word> */
                if( num_paren == 0){
                  return token;
                } else {
                  String return_value =
                     token.concat(ParseExpression(st,num_paren,false,token));
                  if(isDelimiter(last_token) || isWhiteSpace(last_token)){
                     return return_value;
                  } else {
                     return (" " + return_value);     }    }       }
            return "";}      }    }
              return "";  }

  protected String LengthDelimitedString(StringTokenizer st, int num_paren){
    String answer = "";

    /* get the length */
    String length_str = st.nextToken();
    Integer length = new Integer(length_str);
    /* add it on */
    answer = answer.concat(length_str);
    /* get the " */
    answer = answer.concat(st.nextToken());
    /* get each character in the string */
    for(int i = 0; i < length.intValue(); i++){
      answer =
answer.concat(st.nextToken("abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890(){}[]<>=+-
*\\/&^~\'`\"_@#$%:;,.!?\t\n\r "));
    }
    if(num_paren == 0){
      return answer;
    } else {
      return(answer.concat(ParseExpression(st,num_paren,false,answer)));    } }

  protected String getNonDelimiter(StringTokenizer st){
    String token = st.nextToken("\n\t\r\"\'()' #");
       //System.out.println("token1first="+token);
         while(isDelimiter(token) && st.hasMoreTokens()){
      token = st.nextToken();
           //System.out.println("token="+token);    }
    if(isDelimiter(token)){
      token = null;    }
    //System.out.println("tokenreturn="+token);
        return token;  }

    protected String getNonWhiteSpace(StringTokenizer st){
    String token = st.nextToken("\n\t\r\"\'()' #");
    while(isWhiteSpace(token) && st.hasMoreTokens()){
      token = st.nextToken();          }
    if(isWhiteSpace(token)){
      token = null;    }
           return token;  }

    protected boolean isDelimiter(String token){
    boolean answer = false;
    if(token.equals(" ") ||
       token.equals("\n") ||
       token.equals("\t") ||
       token.equals("\r") ||
       token.equals("\"") ||
       token.equals("\'") ||
       token.equals("`") ||
       token.equals(",") ||
       token.equals("(") ||
       token.equals(")") ||
       token.equals("#")){
      answer = true;    }
    return answer;  }
```

```
    protected boolean isWhiteSpace(String token){
      boolean answer = false;
      if(token.equals(" ") ||
         token.equals("\n") ||
         token.equals("\t") ||
         token.equals("\r")){
        answer = true;    }
      return answer;  }

      public Enumeration getFields() {
      return field_value_pairs.keys();  }

    static public boolean testValidity(KQMLmessage message) {
      boolean validity = false;
      if( message.field_value_pairs.containsKey("performative") &&
//        message.field_value_pairs.containsKey("sender") &&
//        message.field_value_pairs.containsKey("receiver") &&
//        message.field_value_pairs.containsKey("language") &&
//        message.field_value_pairs.containsKey("ontology") &&
          message.field_value_pairs.containsKey("content")) {
        validity = true;    }
      return validity;  }

    public void addFieldValuePair(String field, String value) {
      //System.out.println("value="+value);
//        System.out.println("field="+field);
          field_value_pairs.put(field, value);  }

    public String getValue(String field) {
      String value = null;
      if(field_value_pairs.containsKey(field)){
        value = (String)field_value_pairs.get(field);    }
      return value;  }

    public String getReadString() {
      String answer = "";
      Enumeration fields = field_value_pairs.keys();
      while(fields.hasMoreElements()){
        String field = (String)fields.nextElement();
        answer = answer.concat(field + ": " + getValue(field) + "\n");    }
      return answer;  }

    public String getSendString () {
      String answer = "(" + getValue("performative");
      Enumeration fields = field_value_pairs.keys();

      while(fields.hasMoreElements()){
        String field = (String)fields.nextElement();
        if(!field.equals("performative")){
          answer = answer.concat(" :" + field + " " + getValue(field));      }    }
      answer = answer.concat(")");
      return answer;  }}
/**************************************/
package java_kqml;
/*
*bilingual translator
*language
*/

class Language {
  public Language(){ }
  public Language(String input){
    this();
  parseString(input); }
  public void parseString(String input){   } }
```

APPENDIX (D)

Coordination Agents

```
/*
 *
 * Control
 *
 */
import java_kqml.*;
import java.awt.*;
import java.applet.*;
import class.cm.*;
import java.util.StringTokenizer;
import java.lang.Thread;

public class Control extends Applet{

//Interface elements
        Button runButton = new Button("Click here to start grapes diagnosis ES");
//Parameter declarations
        private String hostName;
        private String portNoSt;
        private int portNo;
        private int iNo;
        private String thostName;
        private int tportNo;
        private String [] portArray;
        private String GrowthSt;
        public AnserArrayOBJ AnserArray;
        GetConnection cl;

        private final String PARAM_host = "dhost";
        private final String PARAM_port = "dport";
        private final String PARAM_thost = "thost";
        private final String PARAM_tport = "tport";

//HC Declaration
        HCNodes ex = new HCNodes();

public String getAppletInfo()
        {return "Name: grapes diagnosis ES\r\n" +                    " ;       }

        public String[][] getParameterInfo()
        {       String[][] info =
                {
                        { PARAM_host, "String", "" },
                        { PARAM_port, "int", "" },
                        { PARAM_thost, "String", "" },
                        { PARAM_tport, "int", "" },        };
                return info;                        }

        public void       init()[
        String param;
        param = getParameter(PARAM_host);
        if (param != null)
                hostName = param;

        param = getParameter(PARAM_port);
        if (param != null)
        //portNo = Integer.parseInt(param);
        portNoSt = (param);
        param = getParameter(PARAM_thost);
        if (param != null)
                thostName = param;
        param = getParameter(PARAM_tport);
        if (param != null)
        tportNo = Integer.parseInt(param);
        setFont(new Font("Helvetica", Font.PLAIN, 14));
        add(runButton);}

        public void run() {
        String [] portArray= parseStr(portNoSt,",");
        int NPort=portArray.length;
        //AnserArray= new String[NPort];
        ex.HcCkb.GDB.reset();
        runButton.disable();
        ex.run(); //run HC
        String out = ex.HcCkb.GDB.getObjAll();
//      System.out.println("OUT="+out);
        //collectSt=out;
        if(out.equals("[null-disorder_exist-no]"));
        else
        {
        out = "'(diagnosis :language prolog :content "+out+". )'" ;
        System.out.println("STARTED EXECUTION OF COORDINATION AGENT AT DIAGNOSIS SESSION");
        System.out.println("COORDINATION OUTPUT: "+out);
        GrowthSt= ex.HcCkb.growth_stage.getValue();
```

```
            AnserArray = new AnserArrayOBJ();
            for(int i=0;i<NPort;i++){
            portNo = Integer.parseInt(portArray[i]);
            iNo=i;
             cl = new GetConnection(hostName,portNo,out,AnserArray,iNo);
                    cl.start();
                    try {cl.join();}catch(InterruptedException e){}
                   // System.out.println("after join="+AnserArray.ANSER[0]);     }
            String contents="";
            for(int i=0;i<AnserArray.ANSER.length;i++){
            //System.out.println("Answering *********="+AnserArray.ANSER[i]);
            XMLmessage message = new XMLmessage(AnserArray.ANSER[i]);
            String performative = message.getValue("performative");
            String content = message.getValue("content");
            //System.out.println("performative is  "+performativo );
            //System.out.println("content is  "+content);
            if(performative.equals("reply"))
            contents =content+","+contents;
            else ;
            }
            //System.out.println("final content is  "+contents);
            if (contents.equals(""))
            //{showStatus("THE SYSTEM WILL RUN TREATMENT SESSION...");
            {showStatus("SORRY I HAVE NO DIAGNOSIS...");
            String sorry = "SORRY I HAVE NO DIAGNOSIS...";
            Err errm =new Err("Diagnosis result", sorry,new Frame());
            errm.setFont(new Font("Helvetica", Font.PLAIN, 12));
            errm.setBackground(Color.white);
            errm.setForeground(Color.blue);
            errm.setLocation(200,200);
            errm.resize(150,100);
            errm.show();
            }
            else DiagReply(contents); }}
    /
    public boolean action(Event evt, Object arg) {
            if (evt.target == runButton)
                    {       run();   }
            /* else if(evt.target == resetButton)
             ex.HsCkb.GDB.reset(); */
           return false; }

      public void start() {};

    public String[] parseStr(String inStr,String delem) {
                    StringTokenizer st = new StringTokenizer(inStr,delem,false);
                    String[] newList = new String[st.countTokens()];
                    int  count = 0;
                    while (st.hasMoreTokens()) {
                            newList[count] =st.nextToken();
                            if (newList[count].equals("{}"))
                                    newList[count]="";
                    //System.out.println("the output content is "+newList[count] );
                            count ++;                       }
                            return newList;   }

    /* ------------------------- */

    class GetConnection extends Thread {
        private String Shost;
        private int Sport;
             private int i;
             private String output;
             private AnserArrayOBJ AnserArrayr;

        public GetConnection(String  host, int port,String out,AnserArrayOBJ AnserArrayy,int iNo) {
            Shost = host;
            Sport = port;
                    output = out;
                    i=iNo;
                    AnserArrayr = AnserArrayy; }

        public void run() {
        try {

             ESClient esC =  new ESClient(Shost,Sport);
             //System.out.println("Cordination output="+output);
             System.out.println("OUTPUT TO HOST:"+Shost+"  PORT:"+Sport);
             //System.out.println("thread port="+Sport);
             //System.out.println("thread i="+i);
             esC.out(output);
             String in =esC.in();
             System.out.println("OUTPUT FROM PORT:"+Sport+ "********>>"+in);
             esC.out("bye");
```

```
aaC.CloseClient();
AnswerArray[i].putAns(it,1);
//System.out.println("end of thread ----------");

        } catch (NullPointerException e) {}
    }
}
```