References

[1] Chapman, V., Hunicke, Robin., 2003, AI for Dynamic Difficulty Adjustment in Games, International conference on advances in computer entertainment technology, pp 429-433, ISBN.1-59593-110-4, Northwestern University.

[2] Chen, X., Gao, Y., Wang, H., 2010, RL-DOT: A Reinforcement Learning NPC Team for Playing Domination Games , IEEE Transactions on computations on computational intelligence and AI in games, Vol. 2, No. 1.

[3] Chen, Y. et al, 2009, To Create Intelligent Adaptive Neuro-Controller of Game Opponent from UCT-Created Data, Sixth International Conference on Fuzzy Systems and Knowledge Discovery.

[4] Cho, S., Jang, S., Yoon, J., 2009, Optimal strategy selection of non-player character on real time strategy game using a speciated evolutionary algorithm, CIG'09 Proceedings of the 5th international conference on Computational Intelligence and Games, pp 75-79.

[5] Cunningham P , Fagan M, Fairclough C, Namee B M, 2001, Research Directions for AI in Computer Games, Trinity College Dublin, Department of Computer Science, TCD-CS-2001-29, pp 5-12

[6] Dignum, F., Westra, J., 2009, Evolutionary Neural Networks for Non-Player Characters in Quake III , IEEE Symposium on Computational Intelligence and Games, pp.302-309.

[7] Doki, K. et al, 2010, Estimation of Next Human Behavior and its Timing for Human Behavior Support, International Conference on Control-Automation-Robotics and Vision, pp.952-957.

[8] Doki, K., et al, 2010, Modeling and recognition method of human behaviors with multi-dimensional time series data, Systems Man and Cybernetics (SMC), pp. 2058-2063.

[9] Feng, S., Tan, A., 2010, Self-Organizing Neural Networks for Behavior Modeling in Games, International Joint Conference on Neural Networks, pp.1-8.

[10] Hao, Y., 2010, Verifying Adaptation of Neuro-controlled Game Opponent by Cross Validation under Supervised and Unsupervised Player Modeling, Software Engineering and Data Mining (SEDM), pp 172-177.

[11] Hashimoto, K., Doki, K., Doki, S., Okuma, S., (2009), Study on modeling and recognition of human behaviors by If-Then-Rules with HMM, Industrial Electronics IECON '09, pp. 3410-3415.

[12] Hong, Y., Liu, Z., 2010, A Preliminary Research on Decision Model Based on Bayesian Techniques For A NPC in Computer Games, International Symposium on Computational Intelligence and Design.

[13] Hughes, H., 2012, "Real lessons from virtual battle", [online], available:http://news.bbc.co.uk/ 2/hi/uk_news/ 7587238.stm.

[14] Lee, B., et al, 2008, An Intelligent NPC Framework for Context Awareness in MMORPG, International Conference on Convergence and Hybrid Information Technology.

[15] Lim, C., Baumgarten, R., Colton, S., 2010, Evolving Behaviour Trees for the Commercial Game DEFCON, Computational Creativity Group, Department of Computing, Imperial College London.

[16] Long, E., 2007, Enhanced NPC behavior using goal oriented action planning, University of Abertay Dundee, pp 14-16.

[17] Mozgovoy M, Rogers.P.C, 2012, Umarov I, Believable and Effective AI Agents in Virtual Worlds, International Journal of Gaming and Computer Mediated Simulations, Vol.4 Issue 2, pp.37-59.

[18] Parsons, T., Reinebold, J., 2011, Neuroscience and Simulation Interface for Adaptive Assessment in Serious Games, IEEE International Games Innovation Conference, pp.93-96.

[19] Rabiner, L.R., 1989, A tutorial on Hidden Markov Models and selected applications in speech recognition.

[20] Shi, Z. et al, 2010, Automatic Game AI Design by the Use of UCT for Dead-End, Sixth International Conference on Natural Computation.

[21] Umarov, I., Mozgovoy, M., Rogers, C., 2012, Believable and Effective AI Agents in Virtual Worlds : Current State nd Future Perspectives, International Journal of Gaming and Computer-Mediated Simulations, Vol. 4(2), pp. 37-59.

[22] Wang, H., Gao, Y., Chen, 2010, X., RL-DOT: A Reinforcement Learning NPC Team for Playing Domination Games, IEEE Transactions on computational intelligence and AI in games, Vol. 2, No. 1, pp.17-26.

[23] White, W. et al, 2007, Scaling Games to Epic Proportions, Cornell University.

[24] Yoo, K., Lee, W., 2008, An Intelligent Non Player Character based on BDI Agent, International Conference on Networked Computing and Advanced Information Management, Vol. 2, pp. 214-219.

Appendix A:

RequestAgent java class

```
import comms.Manager;
import jade.content.lang.sl.SLCodec;
import jade.content.onto.basic.Action;
import jade.core.AID;
import jade.core.Agent;
import jade.core.ContainerID;
import jade.core.behaviours.CyclicBehaviour;
import jade.core.behaviours.TickerBehaviour;
import jade.domain.DFService;
import jade.domain.FIPAAgentManagement.DFAgentDescription;
import jade.domain.FIPAAgentManagement.ServiceDescription;
import jade.domain.FIPAException;
import jade.domain.FIPANames;
import jade.domain.JADEAgentManagement.CreateAgent;
import jade.domain.JADEAgentManagement.JADEManagementOntology;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.MessageTemplate;
import jade.proto.AchieveREInitiator;
import java.awt.Color;
import java.util.ArrayList;
import javax.swing.JTextPane;
import javax.swing.text.AttributeSet;
import javax.swing.text.SimpleAttributeSet;
import javax.swing.text.StyleConstants;
import javax.swing.text.StyleContext;

/**
 *
```

```
 * @author Susantha
 */
public class RequestAgent extends Agent {

   NewJFrame mainPanel;
   int numberOfResourceAgents;
   ArrayList<ResourceAgent> resouceList;
   TrainManager tm;
   private AID[] resourceAgents;
   private NextActionRequest request;
   private boolean isMessageSpaceUsed = false;
   private Position target = null;

   @Override
   protected void setup() {
      System.out.println("Resource Agent Manager created...");

      /**
       * ************************************
       * Initiate TrainManager and GUI
       * ************************************
       */
      Manager.getInstance().Initialize();
      tm = new TrainManager(this);
      Manager.getInstance().setTrainManager(tm);

      try {
         for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
               javax.swing.UIManager.setLookAndFeel(info.getClassName());
               break;
            }
         }
```

```
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Le
vel.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Le
vel.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Le
vel.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Le
vel.SEVERE, null, ex);
        }
        //</editor-fold>

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {

                mainPanel = new NewJFrame();
                mainPanel.setTrainManager(tm);
                mainPanel.setVisible(true);
            }
        });



        Object[] args = getArguments();
        if (args != null && args.length > 0) {


            String r = (String) args[0];
            numberOfResourceAgents = Integer.parseInt(r);
```

```
        for (int i = 1; i <= numberOfResourceAgents; i++) {
            CreateAgent ca = new CreateAgent();

            ca.setAgentName(String.valueOf(i));
            ca.setClassName(ResourceAgent.class.getName());
            ca.setContainer(new ContainerID("Main-Container", null));
            Action actExpr = new Action(getAMS(), ca);
            ACLMessage request = new ACLMessage(ACLMessage.REQUEST);
            request.addReceiver(getAMS());
            request.setOntology(JADEManagementOntology.getInstance().getName());

            getContentManager().registerLanguage(new SLCodec(),
                FIPANames.ContentLanguage.FIPA_SL);


getContentManager().registerOntology(JADEManagementOntology.getInstance());

            request.setLanguage(FIPANames.ContentLanguage.FIPA_SL);
            request.setProtocol(FIPANames.InteractionProtocol.FIPA_REQUEST);
            try {
               getContentManager().fillContent(request, actExpr);
               addBehaviour(new AchieveREInitiator(this, request) {
                  protected void handleInform(ACLMessage inform) {
                     System.out.println("Agent successfully created");
                  }

                  protected void handleFailure(ACLMessage failure) {
                     System.out.println("Error creating agent.");
                  }
               });
            } catch (Exception e) {
               e.printStackTrace();
            }
```

```
        }
        System.out.println(numberOfResourceAgents + " of agents created");
    } else {
        System.out.println("no resource agents sepcified");
    }

    addBehaviour(new CyclicBehaviour(this) {
        @Override
        public void action() {
            try {

                MessageTemplate mt =
MessageTemplate.MatchPerformative(ACLMessage.PROPOSE);

                ACLMessage response;
                response = myAgent.receive(mt);

                if (request != null) {

                    if (response != null) {
                        String s = getLocalName() + " received from " +
response.getSender().getLocalName() + " : " + response.getContent();
                        addMessage(s);

                        String[] m = response.getContent().split(":");
                        int nextAction = Integer.parseInt(m[0]);
                        double pro = Double.parseDouble(m[1]);
                        if (request != null) {
                            request.setNextAction(nextAction);
                            request.setProbability(pro);
                        }

                        evaluateResourceResponse(nextAction, pro);
```

```
        }
        System.out.println(numberOfResourceAgents + " of agents created");
      } else {
        System.out.println("no resource agents sepcified");
      }

      addBehaviour(new CyclicBehaviour(this) {
        @Override
        public void action() {
          try {

            MessageTemplate mt =
MessageTemplate.MatchPerformative(ACLMessage.PROPOSE);

            ACLMessage response;
            response = myAgent.receive(mt);

            if (request != null) {

              if (response != null) {
                String s = getLocalName() + " received from " +
response.getSender().getLocalName() + " : " + response.getContent();
                addMessage(s);

                String[] m = response.getContent().split(":");
                int nextAction = Integer.parseInt(m[0]);
                double pro = Double.parseDouble(m[1]);
                if (request != null) {
                  request.setNextAction(nextAction);
                  request.setProbability(pro);
                }

                evaluateResourceResponse(nextAction, pro);
```

```
                    System.out.println("Response From" +
response.getSender().getLocalName() + ":" + nextAction + ":" + pro);
                }
            }
        } catch (Exception e) {
            System.out.println("Exception happened in..");
            e.printStackTrace();
        }

    }

    private void evaluateResourceResponse(int nextAction, double pro) {
        if (request != null && request.getProbability() < pro) {
            request.setProbability(pro);
            request.setCurrentAction(nextAction);

        }
    }

    private void addMessage(final String string) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {

                String currentS = mainPanel.getMessageSpace().getText();
                appendToPane(mainPanel.getMessageSpace(), "\n" + string, Color.blue);
            }

            private void appendToPane(JTextPane tp, String msg, Color c) {
                StyleContext sc = StyleContext.getDefaultStyleContext();
                AttributeSet aset = sc.addAttribute(SimpleAttributeSet.EMPTY,
StyleConstants.Foreground, c);

                aset = sc.addAttribute(aset, StyleConstants.FontFamily, "Times New
Roman");
```

```
                aset = sc.addAttribute(aset, StyleConstants.Alignment,
StyleConstants.ALIGN_LEFT);

                int len = tp.getDocument().getLength();
                tp.setCaretPosition(len);
                tp.setCharacterAttributes(aset, false);
                tp.replaceSelection(msg);
            }
        });

    }
});

    addBehaviour(new TickerBehaviour(this, 40) {
      protected void onTick() {
        try {
          searchAllResourceProviders();
          String message = null;
          if (request != null && !request.isHasStartedProcess()) {
            //TrainManager has 50ms timeout , but ResourceAgentManager has only
40ms timeout. It makesure that while ResourceAgentManager
            //process the request, TrainManager will not make the request null.
            if ((System.currentTimeMillis() - request.getTimeStamp()) < 40) {;
              if (request != null) {
                addMessage(getLocalName() + " broadcast :" + "Current Status - "
                    + String.valueOf(request.getCurrentAction()) + ", Observation
Sequence - " + request.getObservationSequence());
              }
              if (request != null) {
                message = String.valueOf(request.getCurrentAction()) + "%";

                message += request.getObservationSequence();
                request.setHasStartedProcess(true);
```

```
                    ACLMessage cfp = new ACLMessage(ACLMessage.CFP);
                    for (int i = 0; i < resourceAgents.length; i++) {
                      cfp.addReceiver(resourceAgents[i]);
                      //System.out.println("Send Message to :"+i);
                    }
                    cfp.setContent(message);
                    cfp.setConversationId("book-trade");

                    myAgent.send(cfp);
                  }
                }
              } else {
              }
            } catch (Exception e) {
              System.out.println("Exception happend. \n");
              e.printStackTrace();
            }
          }

          private void addMessage(final String string) {
            java.awt.EventQueue.invokeLater(new Runnable() {
              public void run() {
                String currentS = mainPanel.getMessageSpace().getText();

                appendToPane(mainPanel.getMessageSpace(), "\n\n" + string,
Color.green);
              }
              private void appendToPane(JTextPane tp, String msg, Color c) {
                StyleContext sc = StyleContext.getDefaultStyleContext();
                AttributeSet aset = sc.addAttribute(SimpleAttributeSet.EMPTY,
StyleConstants.Foreground, c);

                aset = sc.addAttribute(aset, StyleConstants.FontFamily, "Lucida
Console");
```

```
                aset = sc.addAttribute(aset, StyleConstants.FontSize, new Integer("24"));
                aset = sc.addAttribute(aset, StyleConstants.Alignment,
StyleConstants.ALIGN_JUSTIFIED);

                int len = tp.getDocument().getLength();
                tp.setCaretPosition(len);
                tp.setCharacterAttributes(aset, false);
                tp.replaceSelection(msg);
            }
          });
        }
    });
    searchAllResourceProviders();
  }

  public NewJFrame getMainPanel() {
    return mainPanel;
  }

  private void appendToPane(JTextPane tp, String msg, Color c) {
    StyleContext sc = StyleContext.getDefaultStyleContext();
    AttributeSet aset = sc.addAttribute(SimpleAttributeSet.EMPTY,
StyleConstants.Foreground, c);

    aset = sc.addAttribute(aset, StyleConstants.FontFamily, "Lucida Console");
    aset = sc.addAttribute(aset, StyleConstants.Alignment,
StyleConstants.ALIGN_JUSTIFIED);

    int len = tp.getDocument().getLength();
    tp.setCaretPosition(len);
    tp.setCharacterAttributes(aset, false);
    tp.replaceSelection(msg);
  }
```

```
private void searchAllResourceProviders() {
   DFAgentDescription template = new DFAgentDescription();
   ServiceDescription sd = new ServiceDescription();
   sd.setType("Behaviour-Modeling-Rule");
   template.addServices(sd);
   try {
      DFAgentDescription[] result = DFService.search(this, template);
      //System.out.println("Found the following resourceAgents:");
      resourceAgents = new AID[result.length];
      for (int i = 0; i < result.length; ++i) {
         resourceAgents[i] = result[i].getName();
         //System.out.println(resourceAgents[i].getName());
      }
   } catch (FIPAException fe) {
      fe.printStackTrace();
   }
}
```

```
void setRequest(NextActionRequest req) {
   this.request = req;

}

public NextActionRequest getRequest() {
   return request;
}

public Position getTarget() {
   return target;
}

public void setTarget(String target) {
```

```
    this.target = Position.extractPos(target);
  }

  public static int getDirectionToTarget(Position playerPos, Position targetPos) {
    //double theta = (atan2(xtarget-x, ztarget-z))*180/M_PI;
    double theta = Math.atan2(targetPos.getX() - playerPos.getX(), targetPos.getY() -
playerPos.getY()) * 180 / Math.PI;
    if (theta > 0) {
      if (targetPos.getX() - playerPos.getX() >= 0) {
        //first quarter
        if (theta >= 45) {
          return 1;
        } else {
          return 0;
        }
      } else {
        //for the third quarter
        if (theta >= 45) {
          return 5;
        } else {
          return 4;
        }
      }

    } else {
      if (targetPos.getX() - playerPos.getX() >= 0) {
        //second quarter
        if (theta >= -45) {
          return 3;
        } else {
          return 2;
        }
      } else {
        //fourth quarter
```

```
            if (theta >= -45) {
                return 7;
            } else {
                return 6;
            }
        }
    }
}
```

Appendix B:

ResourceAgent java class

```
import be.ac.ulg.montefiore.run.jahmm.Hmm;
import be.ac.ulg.montefiore.run.jahmm.ObservationDiscrete;
import be.ac.ulg.montefiore.run.jahmm.OpdfDiscrete;
import be.ac.ulg.montefiore.run.jahmm.OpdfDiscreteFactory;
import directory.TrainManager.ObservationStatus;
import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.domain.DFService;
import jade.domain.FIPAAgentManagement.DFAgentDescription;
import jade.domain.FIPAAgentManagement.ServiceDescription;
import jade.domain.FIPAException;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.MessageTemplate;
import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.AbstractList;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Susantha
 */
public class ResourceAgent extends Agent {
```

```
Rule r;

@Override
protected void setup() {
    System.out.println("Resource Agent " + this.getAID().getLocalName() + "
created...");

    /**
     * ****************************************
     * First time load the data file and make the rule
     * ****************************************
     */
    try {

        File folder = new File("data");
        File[] listOfFiles = folder.listFiles();



        for (int i = 0; i < listOfFiles.length; i++) {
            String id = ((listOfFiles[i].getName()).replaceAll("ProbabilityMatrices",
"")).replaceAll(".dat", "");
            if (listOfFiles[i].isFile() && (id.compareTo(this.getLocalName()) == 0)) {
                System.out.println("File " + listOfFiles[i].getName() + ":" +
this.getLocalName());

                double[][] transitionProbabilities = new double[8][8];
                double[][] emmitingProbabilities = new
double[8][TrainManager.NUMBER_OF_OBSERVATION_STATES];


                FileInputStream fstream = new FileInputStream("data/" +
listOfFiles[i].getName());
                // Get the object of DataInputStream
                DataInputStream in = new DataInputStream(fstream);
                BufferedReader br = new BufferedReader(new InputStreamReader(in));
```

```
String strLine;
int count = 0;
int count2 = 0;
int currentAction = 0;
int nextAction = 0;
//Read File Line By Line
int stringLine = 0;
while ((strLine = br.readLine()) != null) {
   stringLine++;
   // Print the content on the console
   if (stringLine > 1 && stringLine < 10) // transitionProbabilities
   {
      count++;
      String[] list = strLine.split("\t");
      for (int a = 0; a < list.length; a++) {
         transitionProbabilities[count - 1][a] = Double.parseDouble(list[a]);
      }

   } else if (stringLine > 11 && stringLine < 20) {
      count2++;
      String[] list = strLine.split("\t");
      if (list.length !=
TrainManager.NUMBER_OF_OBSERVATION_STATES) {
         System.out.println("Incompatible observation states..!!");
      }
      for (int a = 0; a < list.length; a++) {
         emmitingProbabilities[count2 - 1][a] = Double.parseDouble(list[a]);
      }
   } else if (stringLine == 20)//Current action
   {
      currentAction = Integer.parseInt(strLine);
   } else if (stringLine == 21)//next Action
   {
      nextAction = Integer.parseInt(strLine);
```

```
                }
            }
            //Close the input stream
            in.close();
            Hmm<ObservationDiscrete<TrainManager.ObservationStatus>> hmm =
buildHmm(transitionProbabilities, emmitingProbabilities, currentAction);
            System.out.println("Resource:" + this.getLocalName() + ":" + currentAction
+ ":" + nextAction);
            r = new Rule(hmm, currentAction, nextAction);
            break;
          }
        }

        registerAsAResourceProvider();

      } catch (Exception e) {//Catch exception if any
        System.err.println("Error: " + e.getMessage());
      }

      addBehaviour(new CyclicBehaviour(this) {
        @Override
        public void action() {
          MessageTemplate mt =
MessageTemplate.MatchPerformative(ACLMessage.CFP);
          ACLMessage msg = myAgent.receive(mt);
          if (msg != null) {
            String[] break1 = msg.getContent().split("%");
            int playerCurrentStatus = Integer.parseInt(break1[0]);
            List<ObservationDiscrete<TrainManager.ObservationStatus>>
observationSequence = TrainManager.getObservationSequenceAsObject(break1[1]);
            if ((r.getCurrentAction()) == playerCurrentStatus) {
              if (observationSequence != null && observationSequence.size() > 2) {
                double tempProbability = 0;
```

```
                    tempProbability = r.getHmm().probability(observationSequence);

                    ACLMessage reply = msg.createReply();
                    reply.setPerformative(ACLMessage.PROPOSE);
                    //message format of [nextAction:Probability]
                    reply.setContent(String.valueOf(r.getNextAction()) + ":" +
String.valueOf(tempProbability));
                    System.out.println("Resource Sends : " + myAgent.getLocalName());
                    myAgent.send(reply);
                }
            }
        }
    }
  });


}

public void setRule(Rule rule) {
    r = rule;
}

private Hmm<ObservationDiscrete<ObservationStatus>> buildHmm(double[][]
transitionProbabilities, double[][] observationProbabilities, int currentAction) {
    Hmm<ObservationDiscrete<TrainManager.ObservationStatus>> hmm = new
Hmm<ObservationDiscrete<TrainManager.ObservationStatus>>(8,
        new
OpdfDiscreteFactory<TrainManager.ObservationStatus>(TrainManager.ObservationStat
us.class));
    //set initial probabilities Pi

    for (int i = 0; i < 8; i++) {
      hmm.setPi(i, 0);
    }
    hmm.setPi(currentAction, 1);
```

```
        //using the recorded data, emmiting probabilities are added
        //i means hidden states. Here observation states are hard coded
        for (int i = 0; i < 8; i++) {
            double[] d = new double[]{observationProbabilities[i][0],
observationProbabilities[i][1],
                observationProbabilities[i][2], observationProbabilities[i][3],
                observationProbabilities[i][4], observationProbabilities[i][5],
                observationProbabilities[i][6], observationProbabilities[i][7],
                observationProbabilities[i][8], observationProbabilities[i][9],
                observationProbabilities[i][10], observationProbabilities[i][11],
                observationProbabilities[i][12], observationProbabilities[i][13],
                observationProbabilities[i][14], observationProbabilities[i][15],
                observationProbabilities[i][16], observationProbabilities[i][17],
                observationProbabilities[i][18]};
            OpdfDiscrete opdf = new OpdfDiscrete(TrainManager.ObservationStatus.class,
d);
            hmm.setOpdf(i, opdf);
        }

        //set transition probabilities to the hmm model
        for (int i = 0; i < 8; i++) {
            for (int j = 0; j < 8; j++) {
                hmm.setAij(i, j, transitionProbabilities[i][j]);
            }
        }
        return hmm;
    }

    public static List<ObservationDiscrete<TrainManager.ObservationStatus>>
getObservationSequenceAsObject(String s) {
        String[] observations = s.split(":");
        List<ObservationDiscrete<TrainManager.ObservationStatus>> list;
        list = new ArrayList<ObservationDiscrete<TrainManager.ObservationStatus>>();
```

```
        for (int i = 0; i < observations.length; i++) {
            TrainManager.ObservationStatus ob =
TrainManager.getObservationStatus(observations[i]);
            if (ob != null) {
                list.add(ob.observation());
            }
        }
        return list;
    }

    private void registerAsAResourceProvider() {
        // Register the book-selling service in the yellow pages
        DFAgentDescription dfd = new DFAgentDescription();
        dfd.setName(getAID());
        ServiceDescription sd = new ServiceDescription();
        sd.setType("Behaviour-Modeling-Rule");
        sd.setName("BehaviourModeling");
        dfd.addServices(sd);
        try {
            DFService.register(this, dfd);
        } catch (FIPAException fe) {
            fe.printStackTrace();
        }
    }
}
```