Fabiola Dąbroś

Inżynieria Obliczeniowa gr.1

Ćwiczenia 10 – serwis yellow pages

Wykonanie ćwiczenia rozpoczęłam od utworzenia klasy agenta o nazwie wykonawca. Agent ten na początku swojego istnienia rejestruje się w serwisie żółtych stron podając typ serwisu POTEGA. Następnie przed usunięciem wyrejestrowuje się z serwisu.

Kod prezentuje się następująco:

```
public class wykonawca extends Agent{

    DFAgentDescription dfd;
    ServiceDescription sd;

    protected void setup() {
        super.setup();

        dfd = new DFAgentDescription();
        dfd.setName( getAID() );
        sd  = new ServiceDescription();
        sd.setType( "POTEGA" );
        sd.setName( getLocalName() );
        dfd.addServices(sd);

        try {
          DFService.register(this, dfd );
        }
        catch (FIPAException e) { e.printStackTrace(); }

    }

    protected void takeDown() {
        try {
              DFService.deregister(this);
            } catch (FIPAException e) {e.printStackTrace();}

    }
}
```

Aby się zarejestrować używamy DFAgentDescription. Musimy odnieść się do agenta dokonującego rejestracji oraz do DFD. Konieczne jest też wychwycenie możliwych wyjątków, takich jak nieprawidłowe DFD czy zduplikowane wpisy.

Określamy typ usługi oraz nazwę usługi jaki lokalną nazwę agenta.

W celu wyrejestrowania korzystamy z metody takeDown – uruchamiana gdy agent umiera.

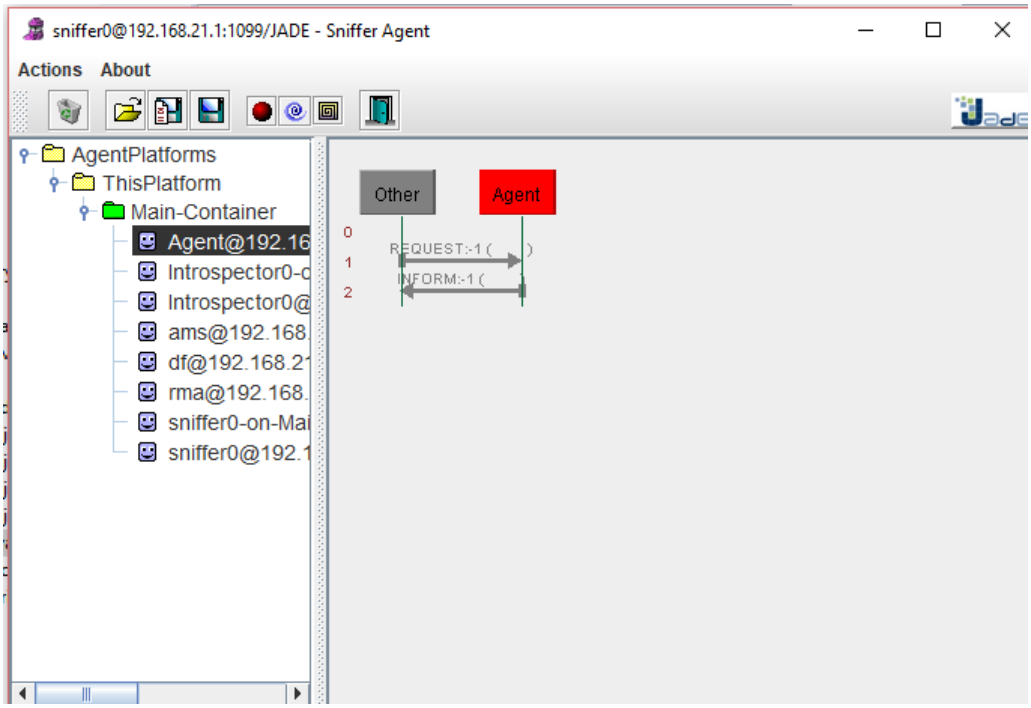
Kolejnym krokiem było zmodyfikowanie klasy agenta o nazwie wykonawca. Jeśli agent otrzyma wiadomość typu REQUEST to parsuje liczbę podaną w treści wiadomości, oblicza wynik podnosząc do drugiej potęgi otrzymana liczbę oraz odpowiada na wiadomość przesyłając wynik wiadomości typem INFORM.

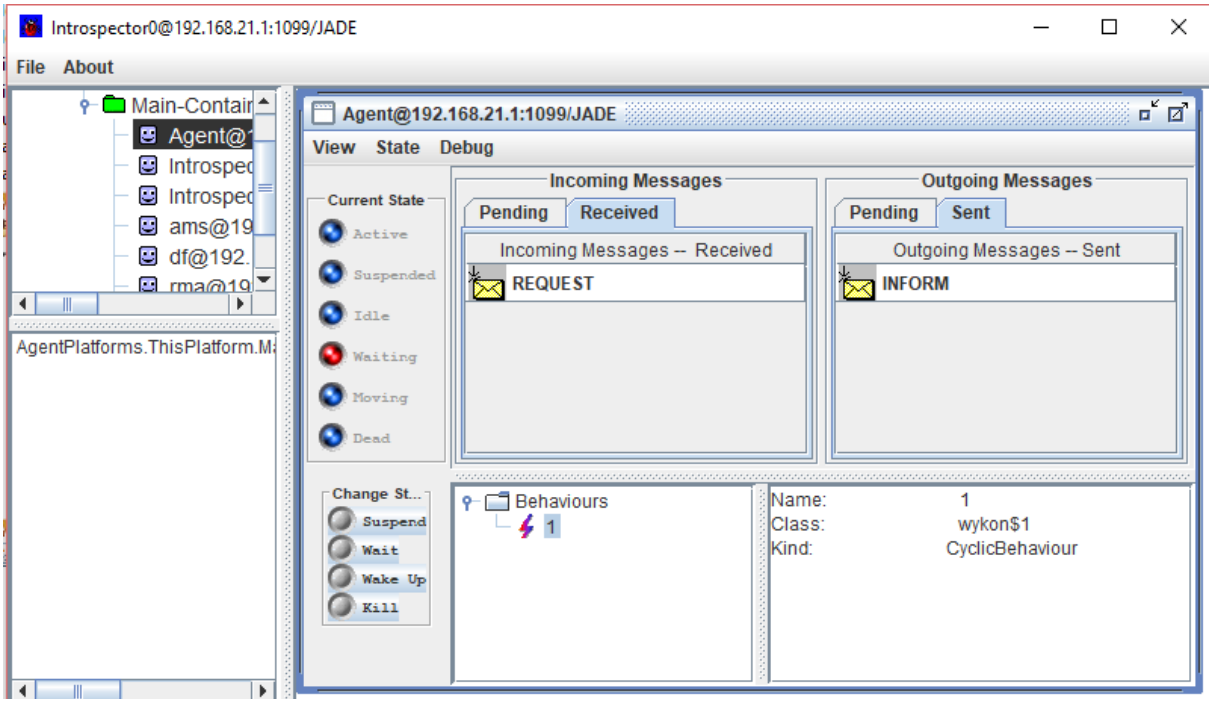
Kod prezentuje się następująco:

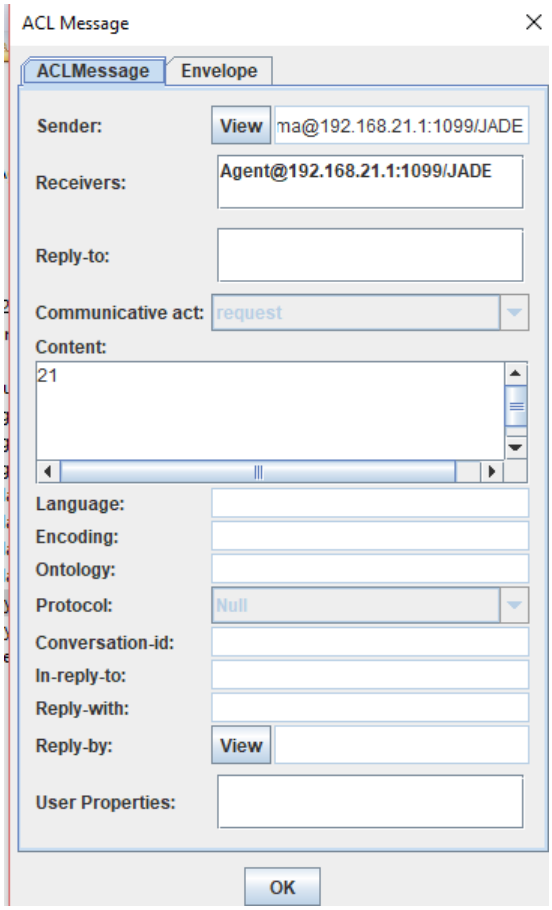
```
MessageTemplate mt = MessageTemplate.MatchPerformative( ACLMessage.REQUEST);
addBehaviour( new CyclicBehaviour()
{
    public void action() {
        ACLMessage msg = blockingReceive(mt);

        if(msg!=null){
            String message = msg.getContent();
            System.out.println("Odebranie wiadomości REQUEST    " + message);

            Double number = Double.parseDouble(message);

            if (message == null){
                System.out.println("Zle dane");
                AID sender = msg.getSender();
                    System.out.println("Wyslanie wiadomości INFORM");
                    ACLMessage response = new ACLMessage(ACLMessage.NOT_UNDERSTOOD);
                    response.addReceiver(sender);
                    response.setContent("Error - wrong data");
                    send(response);
             }
            else if(number != null){
                System.out.println("Liczba z wiadomości po sparsowaniu " + (number));
                number*=number;
                AID sender = msg.getSender();
                System.out.println("Wysłanie wiadomości INFORM: " + Double.toString(number
                ACLMessage response = new ACLMessage(ACLMessage.INFORM);
                response.addReceiver(sender);
                response.setContent(Double.toString(number));
                send(response);
            }
        }
        else{
            block();
```

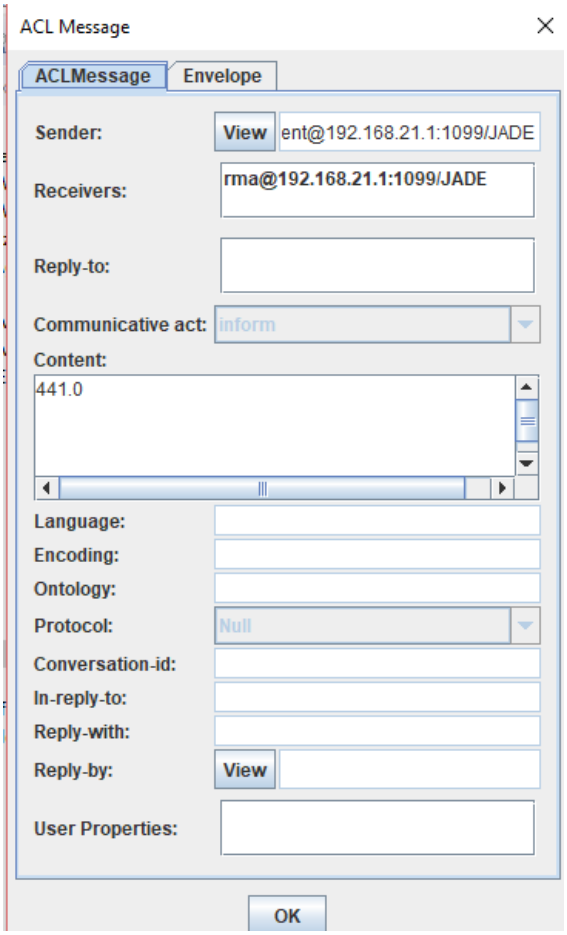
W wyniku czego otrzymujemy:

```
----------------------------------------
Odebranie wiadomości REQUEST    21
Liczba z wiadomości po sparsowaniu 21.0
Wysłanie wiadomości INFORM: 441.0
```









Kolejnym krokiem było utworzenie klasy agenta o nazwie zlecający. Agent ten losuje liczbę całkowitą z przedziału od 0 do 100 oraz co dwadzieścia sekund dowiaduje się w serwisie żółtych stron, którzy agenci oferują serwis typu potęga.
Oprócz tego jeśli jest taki agent, który oferuje ten serwis to nasz agent wysyła do pierwszego agenta z tablicy wyników wiadomość REQUEST zawierającą wylosowaną liczbę oraz odbiera wiadomość INFORM i wypisuję ją na ekranie , po czym usuwa się.

W tym celu należy skorzystać z wyszukiwania w DF. Wyszukiwanie zwraca tablicę, której atrybutu pasują do podanego opisu. Jeśli nie znaleziono pasujących czynników długość tablicy wynosi 0.

Kod prezentuje się następująco:

```
protected void setup() {
    super.setup();

    my = this;
    random = new Random();
    number = null;

    addBehaviour(new OneShotBehaviour(this) {
        public void action() {
            number = (int)(Math.round(random.nextDouble()*100));
            System.out.println("Wysolowana liczba:    " + number);
        }
    });

    FSMBehaviour fsmBehaviour = new FSMBehaviour(this) {
        public int onEnd() {
            myAgent.doDelete();
            return super.onEnd();
        }
    };

    addBehaviour(new TickerBehaviour(this, 2000) {
          protected void onTick() {
              System.out.println("Sprawdzanie usługi POTĘGA");

              DFAgentDescription dfd = new DFAgentDescription();
              ServiceDescription sd = new ServiceDescription();
              sd.setType( "POTEGA" );
              dfd.addServices(sd);

              SearchConstraints ALL = new SearchConstraints();
              ALL.setMaxResults(new Long(-1));
              try
              {
                  DFAgentDescription[] result = DFService.search(myAgent, dfd, ALL);
                  System.out.println("Ilosc uslug:    " + result.length);
                  System.out.println(" ");
                  if(result.length>0) {
                      send(result[0].getName());
                      recieve(this);
                  }
              }
              catch (FIPAException fe) {
                  System.out.println("ERROR"); }
          }
       });
}
private void send(AID aid) {
      ACLMessage msg = new ACLMessage(ACLMessage.REQUEST);
      msg.addReceiver(aid);
      msg.setContent(Integer.toString(number));
      send(msg);
      System.out.println("Wysylanie wiadomości REQUEST przez zlecajacego");
}
private void recieve(Behaviour b) {
     MessageTemplate mt = MessageTemplate.MatchPerformative(ACLMessage.INFORM);
        ACLMessage msg1 = blockingReceive(mt);
        if(msg1!= null) {
            if(msg1.getPerformative() == ACLMessage.INFORM){
                System.out.println("Odebranie wiadomosci inform przez zlecajacego  " +
                my.doDelete();
            }
        }
        else {
            b.block();
        }
```

W wyniku czego otrzymujemy:

```
-------------------------------------------
Wysolowana liczba:   20
Sprawdzanie usługi POTĘGA
Ilosc uslug:    0

Sprawdzanie usługi POTĘGA
Ilosc uslug:    0

Sprawdzanie usługi POTĘGA
Ilosc uslug:    1

Wysylanie wiadomości REQUEST przez zlecajacego
Odebranie wiadomości REQUEST przez wykonawce:    20
Wysłanie wiadomości INFORM przez wykonawce: 400.0
Odebranie wiadomosci inform przez zlecajacego  400.0
Zakonczenie dzialania przez zlecajacego
```