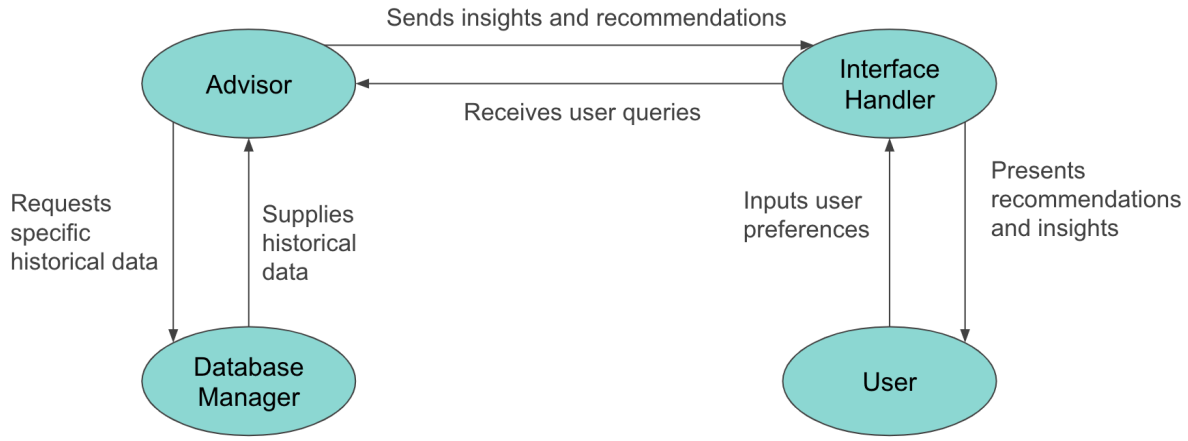
Cristina del Águila Martín. TIES454-2024

DECISION SUPPORT SCENARIO

I have implemented the Decision Support scenario, for this I have 3 agents: the Interface handler Database management and the Advisor agent



1. Interface Handler Agent

- Receives parameters, displays them and sends them to Advisor
- Shows the recommendation received by Advisor

Implementation:

- **setup() method:** the Interface agent indicates its readiness and adds a behavior for receiving messages

```
@Override
protected void setup() {
    System.out.println("Interface agent " + getAID().getName() + " is ready.");
    addBehaviour(new ReceiveMessageBehaviour());
}
```

- **ReceiveMessageBehaviour Class**: this class extends CyclicBehaviour and defines the behavior for continuously receiving messages.
  - **action() method:** the agent receives ACL (Agent Communication Language) messages from other agents.
    - If a message is received and it's not from the Advisor agent, the Interface agent creates a new ACL message of type REQUEST and sends it to the Advisor, containing the content of the received message.
    - If the received message is from the Advisor agent, the Interface agent extracts the recommendation from the message content and displays it.
    - If no message is received or if it's not from the Advisor, the behavior blocks, waiting for the next message

```
private class ReceiveMessageBehaviour extends CyclicBehaviour {

    @Override
    public void action() {
      ACLMessage msg = receive();
      if(msg != null && !msg.getSender().getLocalName().equals("Advisor")){
          ACLMessage request = new ACLMessage(ACLMessage.REQUEST);
          request.addReceiver(new jade.core.AID("Advisor", jade.core.AID.ISLOCALNAME));
          request.setContent(msg.getContent());
          send(request);
      }
      if(msg != null && msg.getSender().getLocalName().equals("Advisor")){
          String recommendation = msg.getContent();
          System.out.println(recommendation);
       }
      else{
          block();
      }
    }
```

2. Advisor Agent:

Its primary function is to handle communication between the Interface agent and the Database agent and to generate a recommendation based on the information.

- Receives the “destination” variable from the Interface
- Send the “destination” variable to the Database agent
- Receive the best “month” for the “destination” from the Database agent
- Generates a recommendation with the “month” and “destination” and sends it to the Interface agent

- **setup() method:** the Advisor agent indicates its readiness and adds a behavior for receiving messages

```
@Override
protected void setup() {
    System.out.println("Advisor agent " + getAID().getName() + " is ready.");
    addBehaviour(new ReceiveMessageBehaviour());
}
```

- **ReceiveMessageBehaviour Class:** extends CyclicBehaviour and defines the behavior for continuously receiving messages.
  - **action() method:** the agent receives ACL (Agent Communication Language) messages from other agents
    - If the received message is from the Interface agent, the Advisor agent extracts the “destination” from the message content and sends it to the Database agent for further processing
    - If the received message is from the Database agent, the Advisor agent processes the received recommendation message, constructs a response, and sends it back to the Interface agent

- If no message is received or if it's not from the Interface or Database agents, the behavior blocks, waiting for the next message

```
private class ReceiveMessageBehaviour extends CyclicBehaviour {
    @Override
    public void action() {
       //El Advisor muestra lo que le ha enviado Interface y lo envía a la Database
       ACLMessage msg = receive();
       if(msg != null && msg.getSender().getLocalName().equals("Interface")){
          String destination = msg.getContent();
          sendToDatabase(destination);
        }
       else if (msg != null && msg.getSender().getLocalName().equals("Database")) {
            String content = msg.getContent();
            String[] params = content.split(" ");

            if (params.length == 2) {
                String month = params[0];
                String destination = params[1];
                String recommendation = "The best month to travel to " + destination + " is " + month;
                sendToInterface(recommendation);
            } else {
                System.out.println("Error: The message does not contain the two expected parameters");
            }
        }else {
           block();
       }
```

- **Helper Methods:**
  - **sendToDatabase(String destination):** sends the “destination” received from the Interface agent to the Database agent for processing

```
private void sendToDatabase(String destination) {
    ACLMessage request = new ACLMessage(ACLMessage.REQUEST);
    request.setContent(destination);
    AID databaseAID = new AID("Database", AID.ISLOCALNAME);
    request.addReceiver(databaseAID);
    send(request);
```

  - **sendToInterface(String recommendation):** sends the recommendation generated by the Advisor to the Interface agent for display

```
private void sendToInterface(String recommendation) {
    ACLMessage reply = new ACLMessage(ACLMessage.INFORM);
    reply.setContent(recommendation);
    AID InterfaceAID = new AID("Interface", AID.ISLOCALNAME);
    reply.addReceiver(InterfaceAID);
    send(reply);
```

3. Database

Its primary function is to handle database operations related to retrieving information about the best month to travel to a particular destination.

  - Receives the “destination” from the Advisor
  - Consult the json file “data.json” in search of the destination to find out which “month” is best
  - Returns to the Advisor agent the month in which it is best to travel to that destination

- **setup() method**: the Database agent indicates its readiness and adds a behavior for receiving messages

```
@Override
protected void setup() {
    System.out.println("Database agent " + getAID().getName() + " is ready.");
    addBehaviour(new Database.ReceiveMessageBehaviour());
}
```

- **ReceiveMessageBehaviour Class**: extends CyclicBehaviour and defines the behavior for continuously receiving messages
  - **action() method:**
    - The agent receives ACL (Agent Communication Language) messages from other agents
    - If the received message is from the Advisor agent, the Database agent extracts the “destination” from the message content and looks up the best month to travel to that destination from a JSON database (data.json)
    - If the best month is found, the Database agent constructs a response message containing the month and destination, and sends it back to the Advisor agent
    - If no message is received or if it's not from the Advisor agent, the behavior blocks, waiting for the next message
    -

```
private class ReceiveMessageBehaviour extends CyclicBehaviour {
    @Override
    public void action() {
       ACLMessage msg = receive();
       if(msg != null && msg.getSender().getLocalName().equals("Advisor")){
          String destination = msg.getContent();
          bestMonth(destination);
        }
       else{
           block();
       }
```

- **Helper Methods:**
  - **bestMonth(String destination):** retrieves the best month to travel to the specified destination from a JSON database

```
private void bestMonth(String destino) {
    JSONParser jsonParser = new JSONParser();

    try (FileReader reader = new FileReader("/Users/cristinadelaguilamartin/NetBeansProjects/Agents/src/main/ja
        Object obj = jsonParser.parse(reader);
        if (obj instanceof JSONObject) {
            JSONObject data = (JSONObject) obj;
            String month = (String) data.get(destino);  // "destino" is the key for the month

            if (month != null) {
                sendToAdvisor(month, destino);
            } else {
                System.out.println("Destino no encontrado");
            }
        } else {
            System.out.println("Error: el archivo JSON no tiene el formato esperado");
        }
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (ParseException e) {
        e.printStackTrace();
    }
}
```

- **sendToAdvisor(String month, String destination):** sends the best month and destination information to the Advisor agent

```
private void sendToAdvisor(String month, String destination) {
    ACLMessage reply = new ACLMessage(ACLMessage.INFORM);
    String content = month + " " + destination;
    reply.setContent(content);
    AID AdvisorAID = new AID("Advisor", AID.ISLOCALNAME);
    reply.addReceiver(AdvisorAID);
    send(reply);
}
```

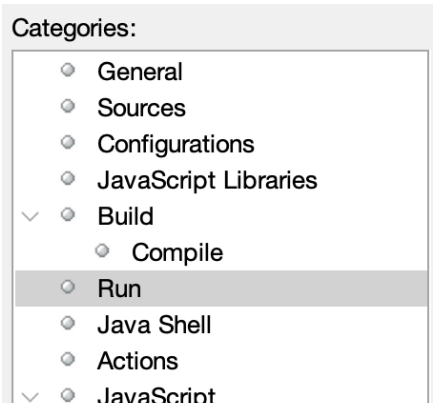
4. data.json

The data.json file contains a JSON object representing information about the best months to travel to various destinations. Each key in the object represents a destination, and its corresponding value represents the best month to visit that destination

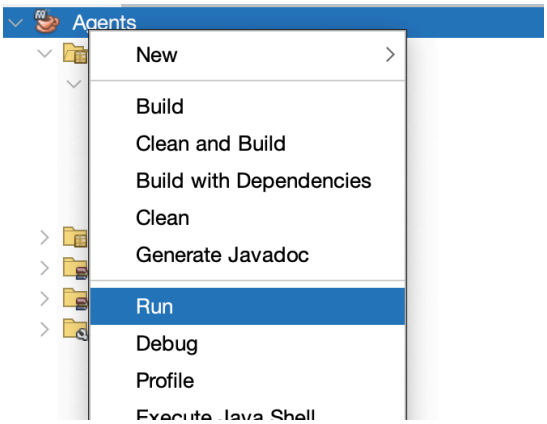

```
{
    "Afghanistan": "January",
    "Albania": "February",
    "Algeria": "March",
    "Andorra": "April",
    "Angola": "May",
    "AntiguaandBarbuda": "June",
    "Argentina": "July",
    "Armenia": "August",
    "Australia": "September",
    "Austria": "October",
    "Azerbaijan": "November",
    "Bahamas": "December",
    "Bahrain": "January",
    "Bangladesh": "February",
    "Barbados": "March",
    "Belarus": "April",
    "Belgium": "May",
    "Belize": "June",
    "Benin": "July",
    "Bhutan": "August",
    "Bolivia": "September",
    "BosniaandHerzegovina": "October",
    "Botswana": "November",
    "Brazil": "December",
    "Brunei": "January",
    "Bulgaria": "February"
```

5. Demonstration

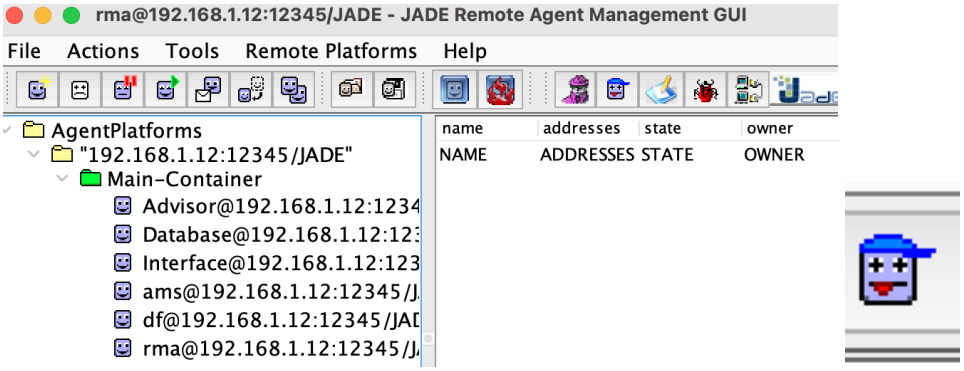
1. Configuration



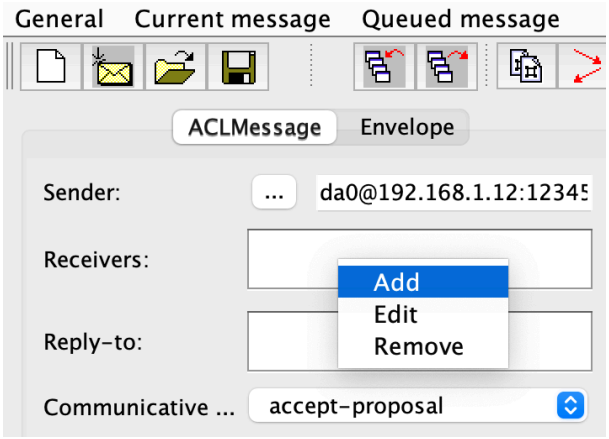
2. Run the project



3. In the GUI, go to “Start DummyAgent”



4. Add Receiver



I add the Interface agent because it is to whom I pass the destination I want to go to

For example, I write write “Andorra”

5. The Interface agent sends the “destination” to the Advisor agent.
   The Advisor agent sends the “destination” to the Database agent.
   The Database agent searches in the data.json file which is the best month to travel for the “destination”
   The Database agent sends the best month and the destination to the Advisor agent
   The Advisor agent generates a recommendation and sends it to the Interface agent
   The Interface agent shows the recommendation

The result for the example “Andorra” is

```
------------------------------------------
Advisor agent Advisor@192.168.1.12:12345/JADE is ready.
Interface agent Interface@192.168.1.12:12345/JADE is ready.
Database agent Database@192.168.1.12:12345/JADE is ready.
The best month to travel to Andorra is April
```

6. Bibliography

- Use json in java:
  - https://www.youtube.com/watch?v=yLf2-r8w9lQ
  - https://es.stackoverflow.com/questions/443785/c%C3%B3mo-leer-correctamente-los-datos-de-un-json-en-java
  - https://www.youtube.com/watch?v=oQAc0N11Cro
- Download json.simple: https://mvnrepository.com/artifact/com.googlecode.json-simple/json-simple/1.1.1
- JADE tutorial