Brokering Semantic Web Agents: A Use Case for a Defeasible Reasoning Service

Kalliopi Kravari, Efstratios Kontopoulos and Nick Bassiliades
{kkravari, skontopo, nbassili}@csd.auth.gr
Dept. of Informatics, Aristotle University of Thessaloniki, GR-54124, Thessaloniki, Greece

This demo presents a complete example of applying defeasible logic in a brokering trade scenario. The application chosen is *apartment renting*[1], an activity that is often tedious and time-consuming. Three independent parties are involved: the *customer*, the *broker* and the *mediator*. The three parties are represented by intelligent agents that intercommunicate, forming a Multi-Agent System (MAS), implemented in JADE[2].

- The *customer* (let's call him Carlos) is actually a potential renter that wishes to rent an apartment based on his requirements (e.g. apartment size, location, floor, facilities etc.) as well as his preferences (e.g. cheapest, largest apartment etc.).
- The *broker*, on the other hand, possesses a number of available apartments stored in an appropriate database. His role is to match Carlo's requirements with the features of the available apartments and eventually propose suitable flats to the potential renter.
- The *mediator* is an independent third-party service that has the capability of processing inference on defeasible logic rule bases and producing the results in the form of an RDF file. The input to the mediator is the rule base and the output is represented by the RDF document.

More specifically, Carlo's requirements are formulated as below:

1. *Carlos is looking for an apartment of at least 45 m^2 with at least 2 bedrooms. If it is on the 3rd floor or higher, the house must have an elevator. Also, pet animals must be allowed.*
2. *Carlos is willing to pay \$300 for a centrally located 45 m^2 apartment, and \$250 for a similar flat in the suburbs. In addition, he is willing to pay an extra \$5 per m^2 for a larger apartment, and \$2 per m^2 for a garden.*
3. *He is unable to pay more than \$400 in total. If given the choice, he would go for the cheapest option. His 2nd priority is the presence of a garden; lowest priority is additional space.*

So, he delegates the transaction to his agent that sends Carlos' requirements to a broker in order to get back all the available houses with the proper properties. These requirements are expressed in defeasible logic, in a RuleML-like syntax. A fragment of the rule base is displayed in Figure 1 (a).

```
<rulebase xmlns:carlo_rb="&carlo_rb;" xmlns:carlo="&carlo;"
 rdf_import=".../carlo_ex.rdf" rdf_export="export-carlo.rdf"
 rdf_export_classes="acceptable">
  <_rbaselab>
    <ind type="defeasible" href="&carlo_rb;">carlo-rules</ind>
  </_rbaselab>
  <imp>
    <_rlab ruleID="r1" ruletype="defeasiblerule">
      <ind href="&carlo_rb;r1">r1</ind>
    </_rlab>
    <_head>
      <atom>
        <_opr><rel href="acceptable"/></_opr>
        <_slot name="apartment"><var>x</var></_slot>
        <_slot name="price"><var>y</var></_slot>
        <_slot name="size"><var>z</var></_slot>
        <_slot name="gardenSize"><var>w</var></_slot>
      </atom>
    </_head>
    <_body>
      <atom>
        <_opr><rel href="carlo:apartment"/></_opr>
        <_slot name="carlo:name"><var>x</var></_slot>
        <_slot name="carlo:price"><var>y</var></_slot>
        <_slot name="carlo:size"><var>z</var></_slot>
        <_slot name="carlo:gardenSize"><var>w</var></_slot>

      </atom>
    </_body>
  </imp>
..............
</rulebase>
```

```
<!DOCTYPE rdf:RDF [
    ...
    <!ENTITY carlo "http://.../dr-device/carlo/carlo.rdf#">
    <!ENTITY carlo_ex "http://lpis.csd.auth.gr/systems/dr-device/carlo/c
]>
<rdf:RDF ... xmlns:carlo="&carlo;" xmlns:carlo_ex="&carlo_ex;">
  <carlo:apartment rdf:about="&carlo_ex;a1">
    <carlo:bedrooms rdf:datatype="&xsd;integer">1</carlo:bedrooms>
    <carlo:central>yes</carlo:central>
    <carlo:floor rdf:datatype="&xsd;integer">1</carlo:floor>
    <carlo:gardenSize rdf:datatype="&xsd;integer">0</carlo:gardenSize>
    <carlo:lift>no</carlo:lift>
    <carlo:name>a1</carlo:name>
    <carlo:pets>yes</carlo:pets>
    <carlo:price rdf:datatype="&xsd;integer">300</carlo:price>
    <carlo:size rdf:datatype="&xsd;integer">50</carlo:size>
  </carlo:apartment>
  ...
</rdf:RDF>
```

Figure 1. (a) Rule base fragment, written in the RuleML-like syntax of DR-DEVICE, and (b) Sample of RDF document displaying available apartments

Then, the first step is done; a document containing Carlos' requirements is send to the broker (broker's agent). The broker has a list of all available houses but does not want reveal it to Carlo, because its one of its most valuable assets. However, the broker cannot process the customer's requirements using defeasible logic, so he requests an independent party's mediator service. The mediator is a special agent that uses the DR-DEVICE[3] de-

[1] The example was taken from : Grigoris Antoniou and Frank van Harmelen, "A Semantic Web Primer", MIT Press, 2004.
[2] JADE – Java Agent Development Environment: **http://jade.tilab.com/**
[3] DR-DEVICE homepage: **http://lpis.csd.auth.gr/systems/dr-device.html**

feasible reasoning engine, in order to infer conclusions from a defeasible logic program and a set of facts in an RDF documents. Hence, the broker sends both the customer's (Carlos') requirements and the list of available houses, waiting to get back the proper houses. Figure 1 (b) displays a part of an RDF document containing the available houses. Consequently, the mediator calls the DR-DEVICE application. The DR-DEVICE processes the above data and returns an RDF document, which contains all acceptable houses that fill all requirements. The mediator sends back the results to the broker and the broker, on his behalf, sends it to the customer. So, steps three and four are already completed.

Eventually, the customer receives the appropriate list and has to decide which house he prefers. The customer agent does not want to send his preferences to the broker, because he is afraid that the broker will take advantage of that and will not present him his most preferred choices. The Carlo's agent sends the list of acceptable houses (an RDF document) and Carlos' preferences (once again in the form of a defeasible logic rule base) to the mediator agent, waiting for reply. Once again, the mediator calls the DR-DEVICE reasoner and gets the most preferred house as an answer. After that, he replies and proposes the best transaction for the customer.

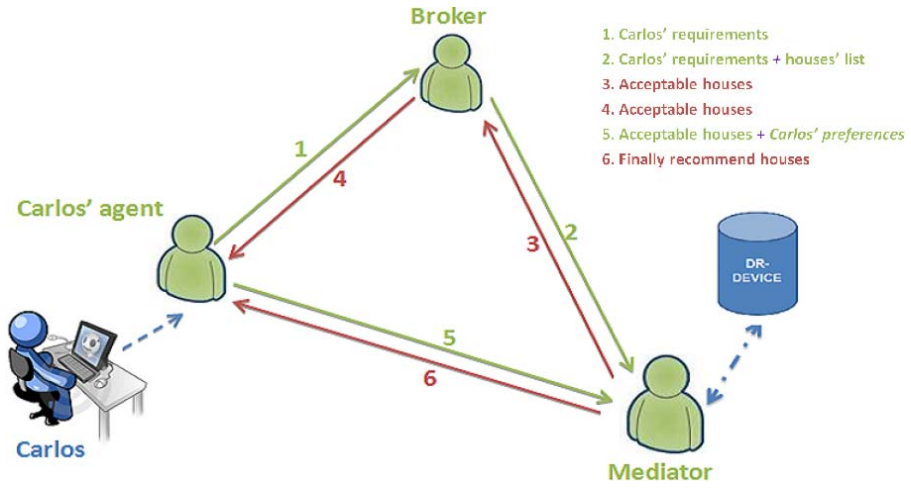


Figure 2. The distinct steps featured in the scenario

At last, the procedure ends and Carlos can safely make the best choice based on his requirements but also his specific preferences. The whole procedure is carried out in six concrete steps, as shown in Figure 2, while Figure 3 displays the JADE sniffer output (message exchange among the agents).

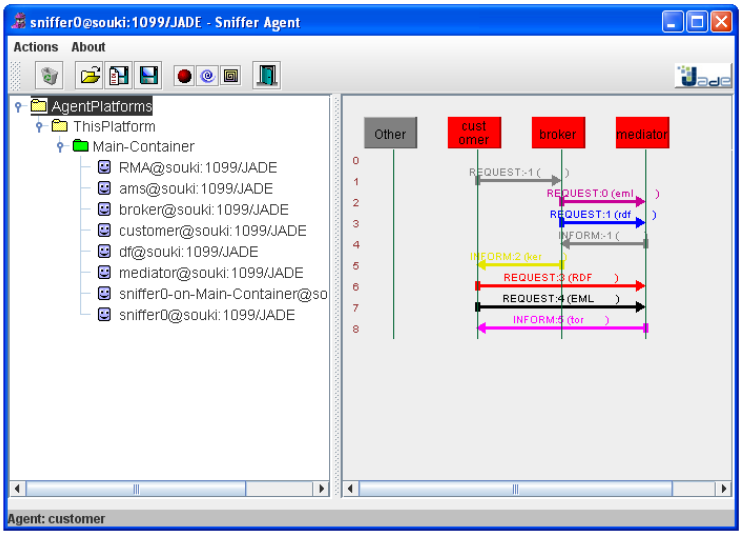
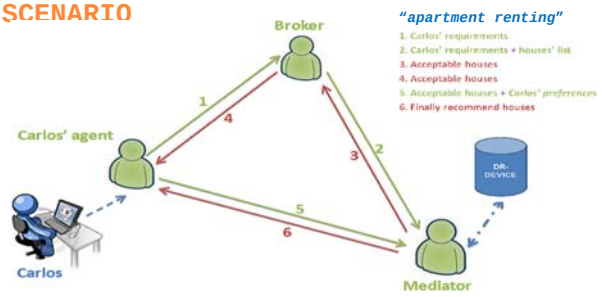


Figure 3. The sniffer window in JADE

BROKERING SEMANTIC WEB AGENTS:
A USE CASE FOR A DEFEASIBLE REASONING SERVICE

THE DISTINCT STEPS FEATURED IN THE SCENARIO



Customer: (Carlos) a potential renter that wishes to rent an apartment based on his requirements and his preferences.

Broker: possesses a number of available apartments stored in an appropriate database.

Mediator: an independent third-party service.

CARLO'S REQUIREMENTS

1. *Carlos is looking for an apartment of at least 45m² with at least 2 bedrooms. If it is on the 3rd floor or higher, the house must have an elevator. Also, pet animals must be allowed.*
2. *Carlos is willing to pay $300 for a centrally located 45 m² apartment, and $250 for a similar flat in the suburbs. In addition, he is willing to pay an extra $5 per m² for a larger apartment, and $2 per m² for a garden.*
3. *He is unable to pay more than $400 in total. If given the choice, he would go for the cheapest option. His 2nd priority is the presence of a garden; lowest priority is additional space.*

```
<!DOCTYPE rdf:RDF [
    ...
    <!ENTITY carlo "http://.../dz-device/carlo/carlo.rdf#">
    <!ENTITY carlo_ex "http://ipls.csd.auth.gr/systems/dz-device/carlo/carlo_ex.rdf#">
]>
<rdf:RDF ... xmlns:carlo="&carlo;" xmlns:carlo_ex="&carlo_ex;">
  <carlo:apartment rdf:about="&carlo_ex;a1">
    <carlo:bedrooms rdf:datatype="&xsd;integer">3</carlo:bedrooms>
    <carlo:centralyes>/carlo:central>
    <carlo:floor rdf:datatype="&xsd;integer">3</carlo:floor>
    <carlo:gardensize rdf:datatype="&xsd;integer">0</carlo:gardensize>
    <carlo:lift>no</carlo:lift>
    <carlo:name>a1</carlo:name>
    <carlo:petsyes>/carlo:pets>
    <carlo:price rdf:datatype="&xsd;integer">300</carlo:price>
    <carlo:size rdf:datatype="&xsd;integer">50</carlo:size>
  </carlo:apartment>
  ...
</rdf:RDF>
```

Available houses are stored in an ***RDF file***

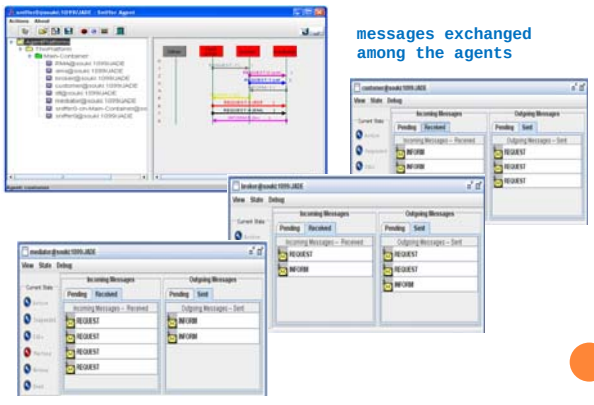
CARLO'S REQUIREMENTS

```
<rulebase xmlns:carlo_rb="&carlo_rb;" xmlns:carlo="&carlo;"
 rdf_import=".../carlo_ex.rdf" rdf_export="export-carlo.rdf"
 rdf_export_classes="acceptable">
  <_rbaselab>
    <ind type="defeasible" href="&carlo_rb;">carlo-rules</ind>
  </_rbaselab>
  <imp>
    <_rlab ruleID="r1" ruletype="defeasiblerule">
      <ind href="&carlo_rb;r1">r1</ind>
    </_rlab>
    <_head>
      <atom>
        <_opr><rel href="acceptable"/></_opr>
        <_slot name="apartment"><var>x</var></_slot>
        <_slot name="price"><var>y</var></_slot>
        <_slot name="size"><var>z</var></_slot>
        <_slot name="gardenSize"><var>w</var></_slot>
      </atom>
    </_head>
    <_body>
      <atom>
        <_opr><rel href="carlo:apartment"/></_opr>
        <_slot name="carlo:name"><var>x</var></_slot>
        <_slot name="carlo:price"><var>y</var></_slot>
        <_slot name="carlo:size"><var>z</var></_slot>
        <_slot name="carlo:gardenSize"><var>w</var></_slot>

      </atom>
    </_body>
  </imp>
.................
</rulebase>
```

Requirements and desires are expressed in ***defeasible logic***, in a ***RuleML-like syntax***.

THE MULTI-AGENT SYSTEM (MAS) IMPLEMENTED IN JADE



messages exchanged among the agents