A Case Study on the Efficacy of Agent UML in Modeling Intelligent Agents

Art Sedighi

Abstract
This paper aims to show that the Agent Interaction Protocol (AIP), a subset of the Agent UML working document can provide a suitable technique for modeling the dynamic aspect of Intelligent Agents. A case study was conducted using thirty software architects fluent in UML to show the clarity of AIP diagrams and the efficacy of the models in conveying the dynamic aspects of Intelligent Agents. In order to conduct this case study, two Intelligent Agents, a Workflow Agent and a Network Monitoring Agent, along with two procedural processes, a Data Logging Facility and a Client Interface Process, are modeled using Agent Interaction Protocols and UML interaction diagrams, respectively.

Background
Intelligent Agents (IA’s) have been used in one form or another in the past to solve some of the problems faced by today’s demand for complex software solutions. Today’s Internet Search Engines and the Simple Mail Transport Protocol (SNMP) represent some the examples of where IA’s have been able to penetrate the mainstream software industry and be integrated into the overall architecture of the enterprise. Many researchers have been surveying the close relation that Object Oriented (OO) techniques share with the methodology that is used for designing agents such as in [6] and [7]. The wide acceptance of OO methodology, techniques and the available tools and standards have made it possible to apply some of those modeling techniques when architecting IA’s for the enterprise. Intelligent Agents (IA’s) thus have become more readily sought after in architecting complex enterprise systems as the technology and expertise for applying IA’s has reached enterprise-level capability, modeling is not quiet there, however. In December 1999, OMG’s Analysis and Design Task Force (ADTF) published a Request for Information (RFI) for possible extensions to the UML 2.0 standard, to enable modeling IA’s in UML [1]. Others have continued this work in this area, such as [2], [3] and [4]. Working documents have been published by a task force created by the some of the OMG members to oversee this effort [5]. Agent UML (AUMU) is still in its developmental stages. Limited applied work has been done, outside the scope of the working group, to study the ease of integrating AUML Interaction Diagrams with UML Interaction Diagrams when modeling an architecture that uses both Agents and Objects as shown in [9]. It is important, just like any other software piece or an application, to be able to model the architecture of an Intelligent Agent, and to be able to incorporate that model into the rest of the enterprise model. As mentioned, there are working groups focused on creating standards to enable Software Architects and Developers to model IA’s in the enterprise. This paper aims to present the efficacy of AUML thru two case studies: modeling of a Network Monitoring Agent, and modeling of a Workflow Agent. These case studies will then be used to determine the ease of readability and clarity of AUML as a framework to model IA’s.

Significance
As per [1], UML does not provide adequate specifications to model IA’s, and the need for such ability has grown due to the increase in demand and complexity of IA’s in the real world. IA’s are becoming part of the overall enterprise-wise software solutions and to do a case study on how the current working documents and standards can benefit both the AUML working committee and will benefit the Software Architects. The AUML working group can use this case study in order to determine if any changes to the draft document is necessarily. The Software Architects can benefit from this study to see what are some of the emerging technologies in this field.

Approach and Methodology
This paper will show the suitability of using Agent Interaction Protocol (AIP), a subset of the AUML working standard, as a modeling technique for modeling the dynamic aspects of IA’s. This measurement will be based on the clarity of the model and its ability of represent an IA or IA’s effectively. A Software Requirement Specification (SRS) document will be used to capture the requirements of the system and will be used again during our evaluation process. The SRS will contain the requirements for the two Agents, a Network-Monitoring Agent (NMA), a simple Workflow Agent (WFA), and the two other software components, a logging facility such as a database that can save the information gathered by Agents, and a simple client application that retrieves information from the Agents in real-time. The SRS will first be reviewed by a panel of 30 software architects fluent in UML, and will be used thereafter for design of the components. That aspect of the process is beyond the scope this research, and it is assumed that the design is complete and has been reviewed and approved by our panel of software architects. The SRS and the design artifacts will then be used as the basis to model our system. The Agents will be modeled using AUML and the logging facility and the client application will use UML for modeling. Each model will focus on the dynamic aspects of the components: Agents and are done using the AIP semantics, and the others are done using UML’s interaction diagram semantics. The models generated will then be incorporated into a mock enterprise-wide architecture of a system. The final model, which will include the two Agents, the logging facility and the client application, and the communication amongst them, along with the other two models will then be evaluated based on the criteria mentioned above. Our panel of software architects will be used for evaluation of the models. The nominal scale of good/medium/bad will be used to gather the responses. A “good” response will mean the AUML is
a good framework to represent IA’s and that the model was very clear in effectively representing the dynamic aspects of the Agent. A “medium” response will mean that the model is not as clear as a UML model, and that the model needs extra description to advocate to the user the architecture of the system. A “bad” response means that AUML is not able to model IA’s clearly. If a “bad” response is given, the reader is asked to comment on the section[s] of the model that is/are not clear.

The model needs to be approved (i.e. a “good” response) by at least 50% of the participants, and disapproved (i.e. a “bad” response) by less than 25% of the participants for the model to be recognized to be as clear to read and understand as a UML interaction diagram.

Requirements

In this section, the requirements of the four processes involved in our research will be given. As mentioned before, two Agents will be modelled in this paper. The NMA and the WFA. A Data Logging Facility (DLF) process and a Client Interface (CIF) process will also be included to show the interaction between the two types.

The NMA and the WFA Agents

In [10], four categories are used to depict Agents: Percepts, Actions, Goals and Environment (PAGE). Humans perceive their environments thru eyes, ears and other organs for sensing, and use legs, hands and other body parts to take actions to achieve a goal. The same analogy is applied to Agents used in our research as shown in figure 1.

For our purposes, both Agents have the complete view of their environment available to them, thus no heuristic search is necessary. NMA is considered to be a Utility Based Agent and the WFA is considered to be a Simple Reflex Agent as per [10].

<table>
<thead>
<tr>
<th>Percepts</th>
<th>Actions</th>
<th>Goals</th>
<th>Env’n</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NMA</strong></td>
<td>Thresholds</td>
<td>Requests data, Sorts data, Checks information, Calculates to get best util func</td>
<td>High utility func</td>
</tr>
<tr>
<td><strong>WFA</strong></td>
<td>Data from NMA</td>
<td>Stores, Notifies, Checks, Calculates</td>
<td>Sort data, Persist data, Inform users</td>
</tr>
</tbody>
</table>

Figure 1. PAGE Analysis of the NMA and the WFA

The WFA must wait for messages such as errors, warning or information messages to arrive from the NMA. The WFA needs to organize the data in manner where the more severe messages (errors for example) get the highest priority. Based on a table or rules, the WFA could take actions to try to correct those errors or warnings. If the data is not of a severe nature such as information messages, the WFA must log the messages into the Data Logging Facility.

The DLF and the CIF Processes

The DLF process is simple database interface process. It takes data from the WFA and it logs that data to the database. DLF also responds to the requests coming from the CIF. The CIF is a process running remotely on the user’s machine, Web driven perhaps that passes the user’s request to the database and displays the retrieved data from the database on the screen after some formatting of the data has been done.

Other Requirements

Other requirements such as Graphical User Interface, the details of implementation such as languages used and reliability and performance of our Agents and processes are beyond the scope of this research and thus are not mentioned here. It is important to note that these requirements play a very important role in the overall architectural design of an application, but for the sake of simplicity and time, we have chosen not to include these requirements in our models.

Agent UML Templates

The AIP calls for the use of Templates as a means to reuse AIP diagrams. Temples are not used here, since they are being evaluated for the final version of UML standard.

Interaction Diagrams

Five interaction diagrams will be shown in this section: one AIP diagram for the NMA, one AIP diagram for the WFA, one AIP that puts the AIP diagrams of the NMA and the WFA together and one UML Interaction Diagram for the DLF and the CIF.

The NMA and the WFA Interaction Diagrams

The AIP diagram of the NMA will be shown first, then the AIP diagram for the WFA. The two AIP diagrams will be put together and presented as one at the end of this section. One needs to keep in mind that the functionalities of the two Agents are cut down significantly and only basic functions are shown in the AIP diagrams.

Figure 2 shows the AIP diagram for the NMA. In this diagram, the NMA is shown to do the following tasks:

- Check the security information of the Host Computers
- Check the return results versus some preset table and issue one of the following replies to the Host Computer:
  - Security Passed if the security information is correct
  - Security Error if the information is flawed or wrong
  - Security Outdated if the information returned by the Host Computer is old and needs to be updated.
• Check the Error Rate of the Network Elements (NE’s) – This task either invoked thru an external process such as the WFA, or an internal timer that is in the WMA invokes it.
  o Check the return results versus some preset Low Water Mark and High Water Mark value and do the following tasks:
    ▪ An OK message is sent to the WFA if the Error Rate is less than the Low Water Mark Value.
    ▪ A Warning message is sent to the WFA if the Error Rate is more than the Low Water Mark, but less than the High Water Mark Value. Along with the Warning message, a Warning Count message is also sent to the WFA. This message contains the number of consecutive counts that have been generated by this node.
    ▪ An Error message is sent to the WFA if the Error Rate is higher than the High Water Mark Value. The Administrator need also be notified, so the NMA sends the content of the problem in another message to the WFA.

The actual task that the WFA takes is beyond the scope of this diagram, and will be depicted later in Figure 3.

Figure 3 shows the AIP diagram for the WFA. In this diagram, the WFA is shown to do the following tasks:
• Re-acts to messages received from the NMA
  o If the message is an OK message, it logs the message with the LoggingFacility.
  o If the message is a Warning message
    ▪ It waits for the Warning Count number to arrive
    ▪ It logs the Warning with the LoggingFacility
    ▪ Depending on the Warning count number, it does one of the following two tasks:
      ▪ It asks the NMA for the information again
      ▪ It generates an internal error and logs that error with the LoggingFacility
  o If the message is an Error message
    ▪ It waits for the EmailAdmin message to arrive from the NMA, which contains the email address information of the administrator for that network
    ▪ It logs the message with the LoggingFacility and it sends an email to the administrator carrying with it the Error message.

Figure 4 shows the integrated Interaction Diagrams for the WFA and the NMA.

### DLF and the CIF Interaction Diagrams

The Interaction Diagram for the CIF and the DLF are rather simple, and will be shown together. Figure 5 shows the Interaction Diagram for the overall interaction of the WFA, the CIF and the DLF. It is important to mention here that the CIF (denoted as ClientInterface in Figure 5) could be a very complicated Web Application with many functions and Actors. In this depiction, only the request relevant to the rest of the subsystems in our research is shown - mainly DisplayLogs.

In this diagram, the CIF and the DLF are shown to do the following tasks:
• The DLF gets a message from the WFA to LogMessage
  o The DLF simply takes the message to be logged, and persists it to the database
• The DLF gets a request from the CIF to DisplayLogs
  o The DLF queries the Database and returns a ResultSet back to the CIF
• The CIF gets requests from the user to DisplayLogs
  o The CIF passes that request to the DLF and awaits a response.
  o When the response comes back from the DLF, the response is displayed to the user

### Results

The results of our survey are presented in this section. Just to recap, a group of thirty software architects fluent in UML were questioned to determine the clarity and efficacy of the Interaction diagrams. Out of the 30 software architects consulted, 19 or 63.3% of the responders responded with a “good” reply meaning that AUML is a good framework to represent IA’s and that the model was very clear in effectively representing the dynamic aspects of the Agent. Furthermore, 7 or 23.3% of the responders responded with a “medium” reply meaning that that the model is not as clear as a UML model, and that the model needs extra description to advocate to the user the architecture of the system. Finally, 4 or 13.3% of the responders responded with a “bad” reply meaning that AUML is not able to model IA’s clearly. Figure 6 further illustrates the outcome of this research.

It is important to mention that almost all responders made their decisions solely based on one factor: amount of information presented by the Agent UML diagrams. If the responds was “good”, the reason was the fact the responder liked seeing more information presented in the diagrams. If the respond was “bad”, the responder did not want to see the diagram be cluttered with more information that it needs to, even if the information is useful and an informative one. A “bad” response was mainly due to the reason that it is not a good idea for the Interaction Diagram to display more information and logic, and that information is actually more confusing.

---

1 The Warning Count message is kept and published by the NMA for two reasons:
   a. Need to show a point in this paper. It is the easiest way to “add intelligence” to the NMA without over complicating it.
   b. Having the NMA publish the Warning count can also be used to determine if a Warning message was lost in-transit for example. This piggy-back can help a “really smart” WFA to take more severe actions in a case where messages are being dropped in the network.
Either an internal timer can trigger this event, or an outside request

Figure 2. The Interaction Protocol diagram of the NMA

Figure 3. The Interaction Protocol Diagram for the WFA

More details shown in Fig. 3

More details shown in Fig. 2
Either an internal timer can trigger this event, or an outside request.
Conclusion

In this research, the readability and clarity of Agent Interaction Protocols, a subset of the Agent UML, was evaluated and confirmed. Two IA’s were modelled along with two procedural processes. The models were presented to a group of software architects, and the responses from the group showed that the new extensions are useful and clear in modelling IA’s.

References