Agent UML Sequence Diagram And Meta-Model
Jinyu Song¹, Zhiyu Zhou², Ying Guan³
¹,² College of Informatics and Electronics
Zhejiang Sci-Tech University Hangzhou China 310018
song_jinyu@126.com
³. Lu He High School,Tongzhou District
Beijing China 101149

Abstract. UML sequence diagrams used to define basic interactions between objects at the level of method invocation while they are not well-suited for describing the complex interactions as they occur in multi-agent systems. In order to solve this problem, Agent UML is proposed based on UML2.0 by OMG and FIPA. Three extensions to UML are analyzed and an example is utilized to show the Agent UML sequence diagram. The major elements of this Agent UML sequence diagram are lifeline, message, execution occurrence, event occurrence, combined fragment and interaction operator. On this condition a draft Agent UML sequence diagram meta-model as a UML class diagram is proposed and the main differences between the UML2.0 meta-model and the draft Agent UML meta-model are researched. Our Agent UML meta-model has a constructive influence on the evolution of Agent UML for the specification of agent interaction protocols.

Keywords: Agent UML, UML2.0, Sequence Diagram, meta-model.

Introduction
Currently, several (formal and semi-formal) techniques exist that describe protocols. These techniques are primarily based on work performed in communication protocol engineering (such as automata or Petri nets) or are specifically designed to agent interaction protocols. Now, agent designers need techniques that formally describe agent-related Interaction Protocol requirements.[1] The Unified Modeling Language(UML)is a visual modeling language for software engineering that has been standardized by the Object Management Group(OMG).while the current UML is sometimes insufficient for modeling agents and agent-based systems. OMG and FIPA are exploring and recommending extensions to UML for the specification of agent interaction protocols(AIP).Agent UML(AUML) based on the OMG’s UML 2.0 specification is designed with this requirement in mind.

The specification of UML 2.0 Interaction diagrams represent a family of diagrams: (1) Sequence diagrams that focus on the sequence of messages between agents, (2) Interaction overview diagrams that offer an overview of the control flow, based on activity diagrams, (3) Communication diagrams that focus on object relationships where the message passing is central, and finally (4) Timing diagrams which are interaction diagrams that show the change in state or condition of a lifeline over linear time.[2]

In this paper, we focus on AUML sequence diagram and major elements of an AUML interaction protocol and then propose a draft AUML sequence diagram meta-model.

AUML Sequence Diagram Extending UML2.0
A UML sequence diagram is a form of interaction diagram which shows objects as lifelines running down the page, with their interactions over time represented as messages drawn as arrows from the source lifeline to the target lifeline. Sequence diagrams are good at showing which objects communicate with which other objects; and what messages trigger those communications.
Usually, one sequence diagram captures the behavior of one use case. These diagrams are mainly used to define basic interactions between objects at the level of method invocation; they are not well-suited for describing complex interactions as they occur in multi-agent systems.

This paper puts emphasis on some possible extensions to UML sequence diagram that can also model agent-based interaction protocols.

**A. extensions to the concept of object.** In UML, role is an instance focused term. In the framework of agent oriented programming by agent-role a set of agents satisfying distinguished properties, interfaces, service descriptions or having a distinguished behavior are meant. The agent lifeline defines the time period during which an agent exists, represented by dotted vertical lines. The lifeline starts when the agent of a given agent role is created and ends when it is destroyed.

**B. extension to message.** Message is a named element that defines one specific kind of communication between lifelines of an interaction in UML 2.0. A message reflects either an operation call or a sending and reception of a signal. While agent has the autonomy, it can not only accept or reject messages, but also can decide whether to run an action based on the contents of the message. So the description which relates to the content of the agent message is more complex than the object. Message notation is extended based on speech-acts theory in AUML to meet requirements of agent system.

**C. extension to UML that support concurrent threads of interaction.** Agent has the ability to handle complex matters, the most typical ability is the concurrent processing to multiple messages or events. Figure 1 shows three recommended extensions that support concurrent threads of interaction.

Fig. 1(a) indicates an and communication: all threads CA-1 to CA-n are sent concurrently. Fig. 1(b) indicates an inclusive or which includes a decision box indicating that a decision box will decide which CAs(zero or more)will be sent. If more than one CA is sent, the communication is concurrent.

Fig. 1(c) indicates an exclusive or, so that exactly one CA will be sent.

Fig. 1 three recommended extensions that support concurrent threads of interaction

For a more detailed discussion of sequence diagrams see Odell [3]and Parunak [4].

Fig.2 an example AUML sequence diagram

1) Lifeline: A lifeline in AUML describes the presence of a participant or a role within the interaction.
2) Message: Messages represent communication between two agents, execution occurrences represent the actions and computations of an agent, and event occurrences represent the events of sending or receiving messages by an agent.

3) Execution Occurrence: Execution occurrences represent the period of time during which an instance is active. An execution occurrence is displayed as a thin vertical rectangle that overlaps the dashed line of a lifeline (see Fig. 2). Because the execution occurrence has duration, it is represented by two event occurrences, the start event occurrence and the finish event occurrence.

4) Event Occurrence: Event occurrence represents moments in time to which actions are associated (see Fig. 2). Event occurrences are ordered along a lifeline. A message has two types of event occurrences, SendEvent and ReceiveEvent. The SendEvent is at the base of message arrow where the message departs from lifeline of the sending object, while ReceiveEvent is at the point of the message arrow where the arrow hits the lifeline of the receiving object.

5) Combined Fragment: Combined fragments divide a sequence diagram into different areas with different Behavior. Each combined fragment has one operator called interaction operator; one or more operands called interaction operands and zero or more guard conditions. Depending on the guard condition, the decision is made on what all operands need to be processed.

6) Interaction Operator: An interaction operator defines how to execute the interaction operands within the context of the combined fragment. Some important interaction operands are alt, opt, loop and break. For a more detailed discussion of Interaction Operator see Cook [5].

**AUML meta-model**

Meta-modeling is a means to define concepts used in a system. This can facilitate analysis and design by identifying activities for instantiating the meta-model entities with respect to the target application. Meta-model identifies which elements should the developer look for, and what relationships and constraints exist for those elements.

The Meta-Object Facility (MOF) is an Object Management Group (OMG) standard for model-driven engineering. The OMG was in need of a meta-modeling architecture to define the UML. MOF is designed as a four-layered architecture. It provides a meta-meta model at the top layer, called the M3 layer. This M3-model is the language used by MOF to build meta-models, called M2-models. The most prominent example of a Layer 2 MOF model is the UML meta-model, the model that describes the UML itself. These M2-models describe elements of the M1-layer, and thus M1-models. These would be, for example, models written in UML. The last layer is the M0-layer or data layer. It is used to describe real-world objects. UML2.0 meta-model defines UML model structure, which is M2 layer. [6]

To date, FIPA has not developed a meta-model for AUML. AUML is based on the interaction protocol diagrams of UML 2.0, for which a meta-model exists. Therefore we propose a draft meta-model for AUML based on a simplified version of the meta-model of UML 2.0. Fig. 3 shows the structure of our AUML meta-model draft as a UML class diagram.

![Fig. 3](structure_of_draft_AUML_meta_model.png)
The main differences between the UML2.0 meta-model and our meta-model draft for AUML are as follows:

1) In our meta-model we have not included the generic meta-classes such as Behaviour, ConnectableElement and NamedElement that are used in UML2.0 as base classes for many different types of diagram element.

2) InteractionOccurrence is omitted for simplicity in this stage of research.

3) The attributes roleName, agentIdentifier and cardinality are added to the class LifeLine due to differences between AUML and UML 2.0.

4) The UML2.0 specification appears to be incomplete, as in the class InteractionFragment a bidirectional association to the class LifeLine is mentioned, but it is not shown in the LifeLine class. Therefore we added the association InteractionFragment–LifeLine to the class LifeLine.

5) In AUML meta-model the class Message is an abstract class. For every FIPA ACL[7] communicative act there is a subclass of message specifying the content structure for that type of message (these subclasses are not shown in figure).

Conclusions

AUML is proposed based on UML2.0 by OMG and FIPA, which is used to model for multi-agent system. AUML sequence diagram and major elements of an AUML interaction protocol are introduced in this paper. The draft meta-model for AUML proposed in this paper is very simple and incomplete. Therefore further research is required to complete the AUML meta-model.

Acknowledgment

This material is based upon work funded by Zhejiang Provincial Natural Science Foundation of China under Grant No. Y1090256.

References