A Methodology for AUML Role Modeling

Mohammad Bakar
Philadelphia University Faculty of Information Technology,
Department of Computer Science
P.O.Box:1 Philadelphia University 19392 Jordan,
Amman, Jordan
mwb_sw@hotmail.com

Said Ghoul
Philadelphia University of Information Technology,
Department of Computer Science
P.O.Box:1 Philadelphia University 19392 Jordan,
Amman, Jordan
sghoul@philadelphia.edu.jo

***Abstract*— an agent can play one or more roles. A role is a specific behavior to be played by an agent, defined in terms of permission, responsibilities, activities, and of its interactions with other roles. An agent plays a role by actualizing the behavior in terms of services to be activated and de-activated in dependence of specific pre-condition and post-conditions. So the need to develop a model representing formally role of an agent and its interaction with other agent is valuable. The need for a role model that expresses how agent assumes and change roles is essential and AUML is not completely supportive. The core parts of AUML are interaction protocol diagrams and agent class diagrams, which are extensions of UML's sequence diagrams and class diagrams, respectively. Agents are assigned to roles, belong to classes and an Interaction Protocol diagram shows interactions between these agents roles along a timeline.**

So, the majority of AUML problems are the lack of: (1) agent role definition methodology, (2) formal semantics to AUML role diagrams, (3) responsibilities (internal) agent role definition (only the external role is defined by sequence diagram), and the agent role control over the time. In this paper, we propose a solution to the above problems. We start by analyzing some significant actual approaches to agent role modeling, we introduce an enhancement to AUML by an agent role definition methodology, and we end by comparing our contribution with similar works.

I. INTRODUCTION

THE concept of an agent has become important in Artificial Intelligence (AI). There are many different usages of the term agent in AI and computer science, and each one appeals to a subtly different notion of agency. Defining agents is not straightforward. There are many opinions [1, 2, 3, 4]. The key characteristics of agents are widely understood to be highly autonomous, proactive, situated, and directed software entities. Other characteristics such as mobility are optional and create a special subtype of agent; whereas some characteristics cannot be used as determining factors since they are really grey shades of a scale that encompasses both objects and agents.

In this context, an autonomous agent is one that is totally independent and can decide its own behavior, particularly how it will respond to incoming communications from other agents. A proactive agent is one that can act without any external prompts. However, it should be noted that this introduces some problems, since there is also a large literature on purely reactive agents that would not be classified as agents with this categorization [1, 4]. Although reactive agents dominate in some domains, in reality, most agents being designed today have both proactive and reactive behavior. Balancing the two is the key challenge for designers of agent-oriented software systems.

Henderson-Sellers and Giorgini in [26] that:" The characteristic of situatedness means that agents are contained totally within some specific environment. They are able to perceive this environment, be acted upon by the environment, and, in turn, affect the environment. Finally, the directedness means that agents possess some well-defined goal and their behavior is seen as being directed towards achieving that goal".

A Multi-Agent System (MAS) [2, 22, 23] is one that consists of a number of agents, which interact each one with another, typically by exchanging messages through some computer network infrastructure. In the most general case, the agents in a multi-agent system will be representing or acting on behalf of users or owners with very different goals and motivation. In order to successfully interact, these agents will thus require the ability to cooperate, coordinate and negotiate with each other, in much the same way that we cooperate, coordinate and negotiate with other people in our everyday lives [2, 5].

Many Agent Oriented (AO) methodologies (e.g. Gaia [6] and Tropos [7]) use the metaphor of the human organization (possibly divided into sub-organizations) in which agents play one or more roles and interact with each other. Human organization models and structures are consequently used to design MAS [2]. Concepts like role, social dependency, and organizational rules are used not just to model the environment in which the system will work, but the system itself. Given the organizational nature of MAS, one of the most important activities in an AO methodology results in the definition of the interaction and cooperation models that capture the social relationships and dependencies between agents and the roles they play within the system. Interaction and cooperation models are generally very abstract, and they

978-1-61284-675-0/11/$26.00

are concretized implementing interaction protocols in later phases of the design.

UML [8] is used in object-oriented systems (OOS) modeling. However, it is not adapted to the modeling of MAS [9, 10]. This is essentially due to the fundamental differences between OOS and MAS. Compared to objects, agents are relatively active and autonomous. Furthermore, objects are reactive while agents are proactive and social. To fill these weaknesses, the UML was extended by the language Agent UML (AUML) [11, 12, 13] supporting new structural elements and diagrams increasing the expression power of the basic language, for tacking into account agent oriented concepts.

An agent can play one or more roles. A role is a specific behavior to be played by an agent, defined in terms of permission, responsibilities, activities, and of its interactions with other roles [17]. An agent plays a role by actualizing the behavior in terms of services to be activated and de-activated in dependence of specific pre-condition and post-conditions. So the need to develop a model representing formally role of an agent and its interaction with other agent is valuable [18, 19]. The need for a role model that express how agent assume and change roles is essential and AUML is not completely supportive [14, 15]. The core parts of AUML are interaction protocol diagrams and agent class diagrams, which are extensions of UML's sequence diagrams and class diagrams, respectively [14]. Agents are assigned to roles belong to classes and an Interaction Protocol diagram shows interactions between these agents roles along a timeline. So, the majority of AUML problems are the lack of: (1) agent role definition methodology, (2) formal semantics to AUML role diagrams [15, 16], (3) responsibilities (internal) agent role definition (only the external role is defined by sequence diagram), and the agent role control over the time.

In this paper, we propose a solution to the above problems. We start by analyzing some significant actual approaches to agent role modeling, we introduce an enhancement to AUML by an agent role definition methodology, and we end by comparing our contribution with similar works.

II. Role Modeling approaches

Roles have been used in MAS extensively, but there is not a methodology or an implemented architecture or framework that satisfies all the requirements or criteria. In the following we consider a case study and some significant formal roles modeling approaches.

A. *Case Study*

While all auction types follow the *M*th- or (*M+1*) st-price rules, the commonly used auction types in today's business are some well known subclasses. Some of them should have the same result in theory but when the bidders are human they usually differ.

English auctions are the most commonly known type of auction, made famous by such auction houses as Sothebys. English auction are first-price, open cry, ascending auctions. The auctioneer starts off by suggesting a reservation price for the good (which may be 0) - if no agent is willing to bid more than the reservation price, then the good is allocated to the auctioneer for this amount. Bids are then invited from agents, who must bid more than the current highest bid - all agents can see the bids being made, and are able to participate in the bidding process if they so desire. When no bidder is willing to raise the bid, then the good is allocated to the bidder that has made the current highest bid and the price he/she pays for the good is the amount of this bid. It turns out that the dominant strategy is for an agent, to successively bid a small amount more than the current highest bid until the bid price reaches their current valuation, and then to withdraw. Agents are human persons. One agent playing an auctioneer role (called auctioneer) and others playing bidder role (called bidder). Each auction involves one Auctioneer and several Bidders.

The Auctioneer Role (Auctioneer). The trading within an Institutional Auction is conducted through the player of this role. It receives instructions to sell, notifies potential buyers and processes their bids according to the auction protocol currently being employed. The auctioneer role is notified at the same time as the winning bidder. An auctioneer (Figure .1) is responsible for offering a product at a particular price to a group of bidder. An auctioneer processes each incoming bid, refusing invalid bids and comparing the bids. Every time a new best bid has been made, all participating bidders agents are informed about the new best bid and asked for new bids. After the auction end time or when no new bid has been made in a defined time frame, all participating bidders agents are notified about the winning bids.

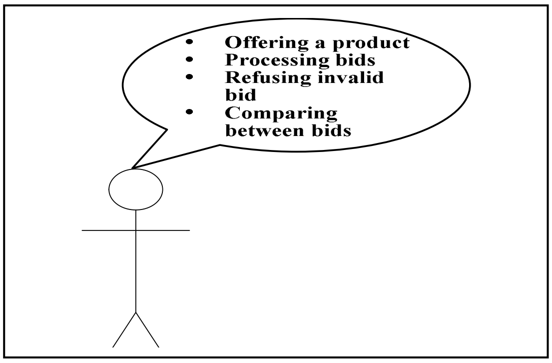

Fig. 1. Auctioneer Role

Bidders (Figure 2) will be capable of proposing bids, receiving information on items available for sale from auctioneer, receiving updates on auction status from auctioneer submitting bids. Proactive bidders will typically use bidding strategies, but this is not assumed.

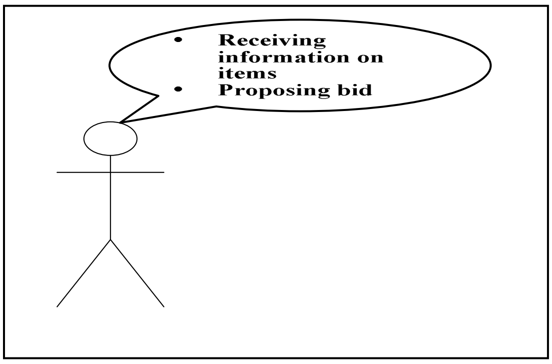

Fig. 2. Bidder Role

B. Agent Role Modeling Approaches

Gaia [6] is a methodology for agent-oriented analysis and design. Its aim is modeling multi-agent systems as organizations where different roles interact. In Gaia, roles are used only in the analysis phase and are defined by four attributes: responsibilities, permissions, activities, and protocols. Gaia exploits a formal notation (based on the FUSION notation) to express permissions of a role. Even if it is not a notation to completely describe roles, it can be helpful because enables the description of roles in terms of what they can do and what they cannot. In addition, Gaia proposes also an interaction model that describes the dependencies and the relationships between the different roles of agents of the system, which it is considered as key point for the functionalities of the system. The interaction model is a set of protocol definitions; each protocol definition is related to a kind of interaction between roles. A roles model is composed of a set of role schema. A role schema draws together the various attributes into a single place as shown in Figure 3 and the Bidder example is shown in figure 4.

Role Schema:	**<name of role>**
Description	short English description of the role
Protocols & Activities	protocols and activities of the role
Permissions	"rights" associated with the role
Responsibilities	
Liveness	liveness responsibilities
Safety	safety responsibilities

Fig. 3. Role Schema

An interaction model is developed based on the role model. It contains a protocol definition for each role in the system. The protocol definition outlines the initiating role, the responding role, the input and output information, as well as a brief description of the processing the initiator carries out during the execution of this protocol. Role model and interaction model in Gaia was designed to handle small-scale, closed systems, however, we identify clearly: (1) An action plan formal model, with responsibilities; (2) a control of these action plans with permission and safety; and (3) a manager of interaction. The formalization and enrichment of these needs and AUML role modeling enhancement with them will be very significant.

Role Schema:	Bidder
Description	this role is capable of proposing bids and check bids.
Protocols and Activities	receive item-info, send bid, check-bid
Permissions	receive item-info//receive information on items send proposal//send bid
Responsibilities Liveness	Bidder = (receive item-info, send bid, check-bid)
Safety	receive item-info = True

Fig. 4. Bidder Role Schema

MASE [20] is a methodology for analyzing and designing heterogeneous multi-agent systems. The final goal of MASE is the automatic generation of code that is correct with respect to the original system specification. A role in MASE is an abstract specification of an entity's expected function that comprises the tasks that should be performed by the role to accomplish its goals. Since each agent class is defined by the set of roles it plays, roles' lifespan equals to the lifespan of the agents in the system. Generally, although there is a one-to-one mapping between roles and agent classes, the designer may combine multiple roles in a single agent class or map a single role to multiple agent classes. Agents, as instances of agent classes, do not deliberate on role assignments and therefore, are not able to revise the devised organization.

In MASE, Roles as shown in figure 5, are denoted by rectangles (e.g. auctioneer and bidder), while the role tasks are denoted by ovals (e.g. Init auction, Send Proposal, etc) attached to the role. Tasks are simply identified in the MASE Role Model. Lines between tasks denote communications protocols that occur between the tasks. The arrows denote the initiator/responder relationship of the protocol with the arrow pointing from the initiator to the respondent. Solid lines (e.g. Initiate-price) indicate peer-to-peer communications, which are generally implemented as external communications protocols. External protocols involve message passing between roles that may become actual messages if their roles end up being implemented in separate agents. Dashed lines denote communication between concurrent tasks within the same role. A lined is dashed (Request set-price) if it will only occur within the same instance of the role in the final system.

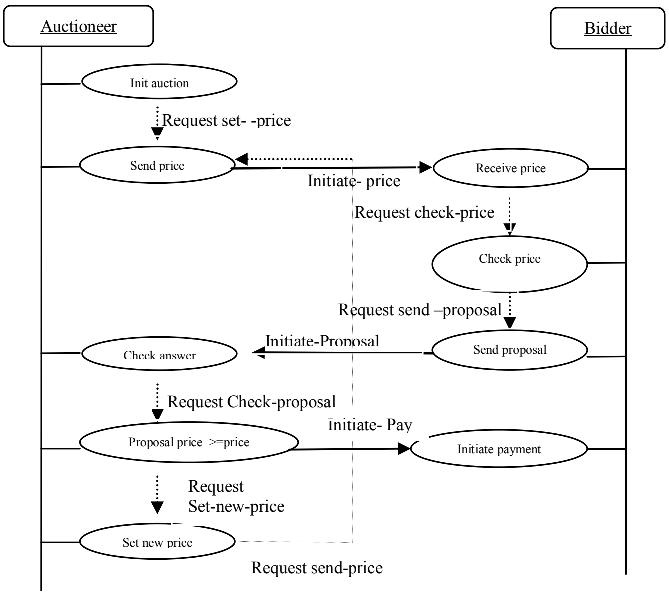

Fig. 5. MASE Role Model For English Auction

In MASE methodology, the responsibilities role of an agent is formalized, the coordination between agents is not represented and the control of the role is not formalized. The AUML role modeling may be enhanced by such responsibilities role formalism.

AMOLA [27] provides the syntax and semantics for creating models of multi-agent systems covering the analysis and design phases of a software development process. It does not only modernize the Gaia2JADE process but offers to the system developer new possibilities compared to other works proposed in the literature [26]. It supports a modular agent design approach and introduces the integrated concepts of intra and inter-agent control. The first defines the agent's role by coordinating his capabilities, while the latter defines the protocols that govern the coordination of the society of the agents. The modeling of the intra and inter-agent control is based on state-charts. AMOLA deals with both the individual and societal aspect of the agents showing how protocols and capabilities can be integrated in agents design. The role model is inspired from GAIA [6] approach, integrating the interaction protocols that agent will be able to participate in (which is a simple service protocol with a requester and responder). The liveness formula is seen as process model that describes the dynamic behavior of the role inside the protocol. It connects all the role's activities using the Gaia operators. The liveness formula defines the dynamic aspect of the role, that is which activities execute sequentially, which concurrently and which are repeating. This role model is then translated to state-charts. The figure 6 shows the role model for the bidder role.

Role: Bidder
Protocols:
Request price: bidder
Respond send propose: auctioneer
Liveness:
Request price = Send price. Wait. Receive price
Respond send propose = (Send propose|| not accept). Receive propose

Fig. 6. Bidder Role Model

AMOLA integrates (without any distinction), formalizes, and controls the responsibilities and coordination role models. However, it does not provide a methodology leading to a complete formal agent role modeling, but an Agent Oriented Software Engineering one. The AUML role modeling may be enhanced by such formalism.

The initial works on AUML [11, 12, 13, 21, 24] focus on agent interaction protocol. Several notions are encompassed in interaction protocol diagrams: Agents roles, messages, constraints on messages and protocol. Marc-Philippe Huget outline in [21] that: "Agents on interaction protocol diagrams can be represented by different manners (1) they are represented by their identity called instance in Agent UML, (2) they are represented by their role in the protocol. For instance, in auction protocols one has usually two roles: the bidder and the Auctioneer, (3) finally, they can be represented by both their identity and their role. The agent class can be given for completing the definition of the roles. Using roles reduce the size of the diagram. Actually, one does not need several identities but just one role if they all share the same role". English auction was taken as case study in [22] (figure 7). Interaction protocol has many drawbacks; the majority of these problems are the lack of formal semantics. However, role modeling in AUML is mainly based on formal definition of interaction between roles of involved agents. The responsibilities role of these agents and their control are not yet formalized, and there is a lack to a role definition methodology. The enhancement of AUML with a formal methodology allowing the definition of agents responsibilities roles, their control and an integration with their interaction protocols will lead to a more complete and useful model.

Several needs, concerning the agent role modeling, come out from the previous study: (1) a methodology for defining agent roles; (2) a model for roles management (defining role, assigning role, un-assigning role and deleting role); (3) a model for agent responsibilities role; and (4) a model for roles control (the control of the responsibilities role and its interaction with others roles) in order to achieve a given goal. Gaia, states clearly responsibilities, cooperation, and control role modeling, but dose not provide a rich and formal model; whereas AMOLA did. MASE focuses mainly on responsibilities role modeling and controller role modeling without a clear separation between them. The most poor is the AUML, which focus only on the interaction protocol modeling and dose not support any other aspect. So, it will be significant to enrich formally the above needs and to integrate them together in AUML.

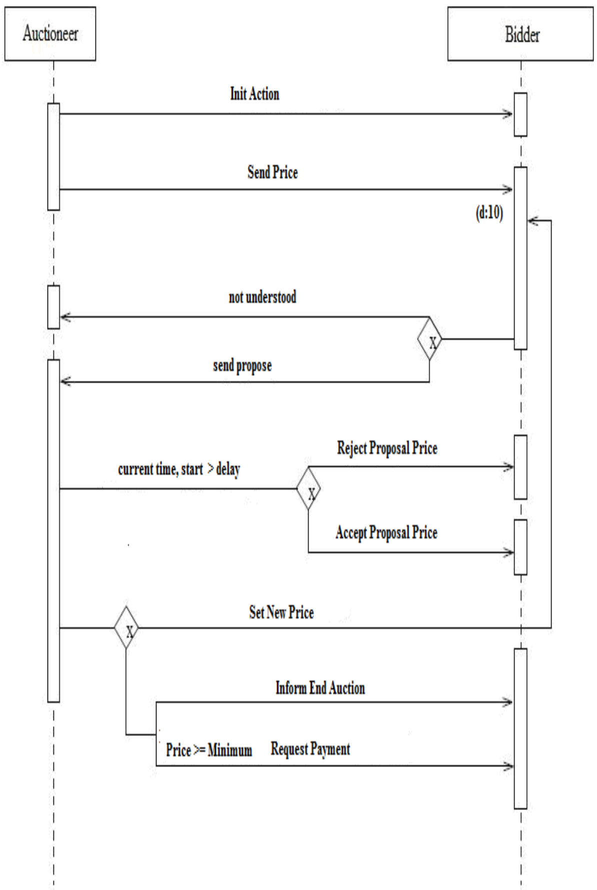

Fig. 7. A General Integrated Approach to Agent Role

III. A METHODOLOGY FOR AUML ROLE MODELING

There is still a lot of terminological and conceptual confusion when organization and related concepts are applied to model MAS. One simple example is the shades of meaning of the notion of organization. What the researchers call organization is something that often varies between two meanings. In one meaning, organization is used to refer to an entity with identity that is represented by a group of agents. In the second meaning, organization is used to refer to constraints (structures, norms and patterns) found in a social context that shape the actions and interactions of agents [24]. In the following (see figure 8), we will present our general integrated methodology to agent roles modeling, its main activities, and coordination rules.

This general approach is largely inspired by the precedent first definition of an organization and is based on an organization agent playing the manager role. This role consists mainly of coordinating involved agents (i.e. employees) roles in order to achieve an organization goal. An involved agent role is composed by two intrinsic sub roles

1) Responsibilities role which is an action plan allowing the agent to achieve its specific goal.
2) Controller role supporting the management of these responsibilities and the management of the agent interaction with others agents roles. By this role we integrate the internal behavior of an agent (responsibilities role) with its external behavior (its interactions with others agents). We remember that only this last behavior was represented by actual AUML with Interaction Protocol.

The methodology supports the following activities:

1) Manager role modeling defining all the involved agents' roles and their control.
2) Involved agents' responsibility roles modeling (their internal behavior).
3) Involved agents' responsibility roles interaction modeling (their external behavior).
4) Involved agents control roles modeling (internal and external behaviors control.

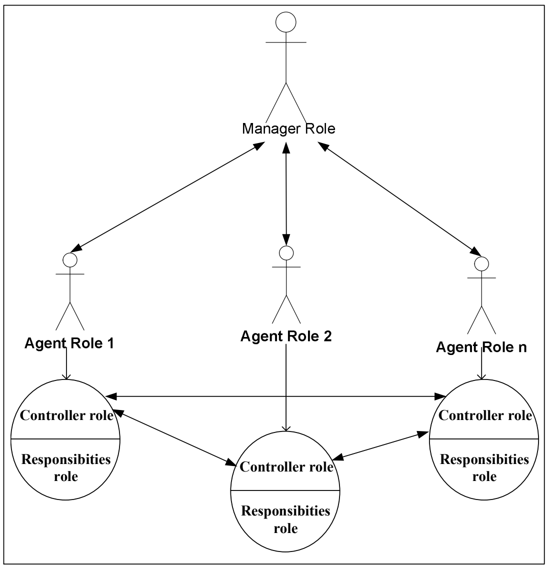

Fig. 8. A General Integrated Approach to Agent Role

Agents' roles are concurrent process and may be formalized by any concurrency theory formalism (Petri-net, Lotos, CCS, CSP …). Our working context (AUML) imposes the use of UML state diagram as formalism for representing our approach [25]. State diagrams consider the different states of a system and how to go from one state to another one through events. Generally State-charts formalism is commonly used in Agent Oriented Software Engineering modeling [28].

Agents' roles are concurrent process and may be formalized by any concurrency theory formalism (Petri-net, Lotos, CCS, CSP …). Our working context (AUML) imposes the use of UML state diagram as formalism for representing our approach [25]. State diagrams consider the different states of a system and how to go from one state to another one through events. Generally State-charts formalism is commonly used in Agent Oriented Software Engineering modeling [28].

The Manager role of an Organization manages and coordinate agents' roles in an organization as shown in figure 9 (e.g. create role, delete role …etc). It assigns new roles to agents, manages interactions with roles through negotiation and/or cooperation to satisfy their design objectives and decides which agents roles must be employed for achieving goal. The management of the agents' interactions at this level (by the manager role) is out of the scope of this work. The observable states of a manager role include:

1) Create role state: in which is created or generated an agent role in the organization.
2) Assign role state: in which is assigned a role to an agent who is able to perform actions and pursue its goals. This assignment implies a direct activation of the role.
3) Unassigned role state: in which the agent leave its role
4) Delete role state: in which is removed or deleted the that the role was already unassigned.

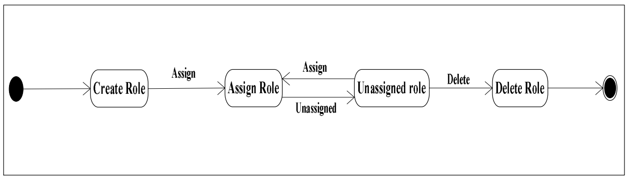

Fig. 9. Manager Role Model

The development of the Manager role model for our case study is shown in figure 10.

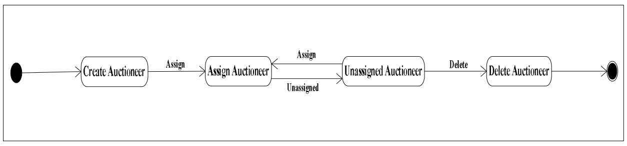

Fig. 10. Manager role model for an English Auction (limited only to auctioneer role)

An "Involved agent" role is composed by two sub roles: Responsibilities Role and Controller Role.

A. Controller Role Model

The involved agents are goal oriented entities that have their controller role model as shown in figure 11. A Controller Role Model manages and controls (internally and externally) the corresponding responsibilities role in order to achieve correctly its goal. It has the following states:

1) Active role state: in which the agent responsibilities role is running.
2) Wait (T) state: in which the agent responsibilities role is waiting for something in time (T).
3) Suspend role state: in which the agent responsibilities role is stopped until an exception happens.
4) Deactivate role state: in which the agent responsibilities role achieved its goals.
5) Synchronize role state: in which the agents' responsibilities roles send messages to others agents' responsibilities roles and wait till those messages are processed prior to keep on running.

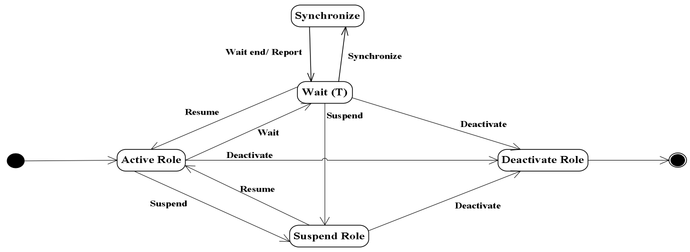

Fig. 11. Controller Role Model

For dealing with interaction with others involved agents role, the synchronize state was introduced to manage the communication via messages exchanging. The message exchange itself is modeled in the synchronization state that controls the sending and receiving of messages. The graphical representation of a synchronization state is shown in figure 12, where arrows on either side denote the control flow of the sender and control flow of the receiver, respectively. Each synchronize state has send transition (send (message, role name)) and the receive transition (receive (message, role name)). When a receiver enters a synchronize state, its controller suspends it until the message has been delivered. This happens whenever the sender reaches synchronize state. After the message has been delivered, the controllers of the receivers resume their corresponding responsibilities roles after the synchronize state. To prevent the controller of the receiver from infinite blocking while waiting a message, an additional (Timeout) transition for each controller of receiver can be attached to the wait (T) state. Whenever the timeout is reached and no message has been delivered in time (T), the controller of the receiver resumes at the state pointed to by the timeout transition.

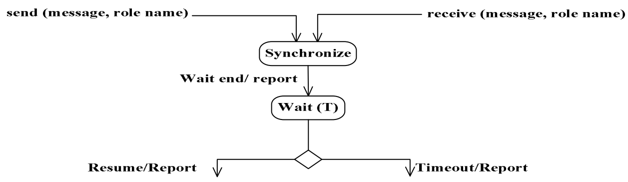

Fig. 12. Synchronization State Model

The controller role model in figure 13 shows how auctioneer controller and bidder controller interact with each others.

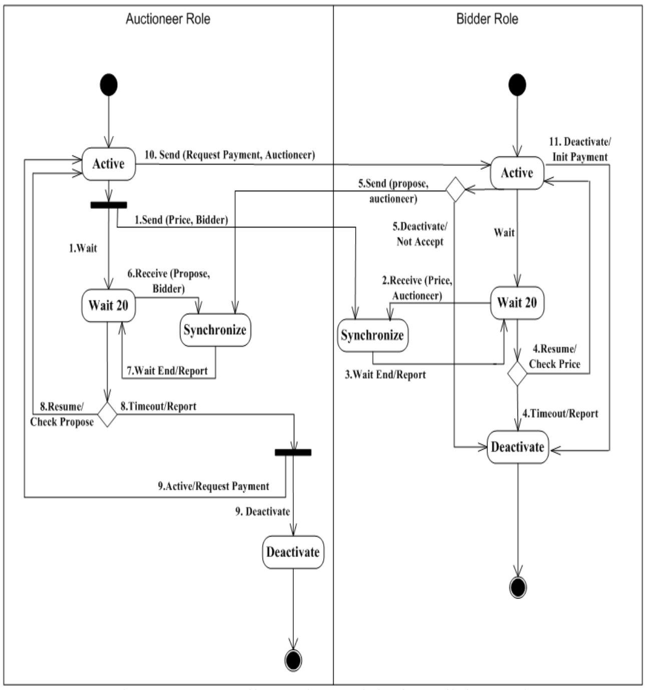

Fig. 13. Controller Role Model of English Auction

B. Responsibilities Role Model.

The responsibilities role is composed of a set of states defining the actions plan (achieving a goal) that should be executed under the control of the associated controller role (figure 14).

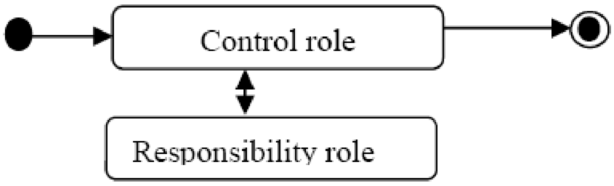

Fig. 14. Responsibility Role Model

The above methodology is supported by an AUML role modeling environment providing tools for modeling the Manager role, Involved agents' responsibility roles, Involved agents' responsibility roles interactions, and Involved agents control roles. The modeling implementation is not challenging by itself, whereas the coherence of the whole integrated methodology models is not obvious. Methodology integrity rules must be identified, formalized, and implemented. The following examples as shown in figure 15 allow a minor evaluation of the problem boundaries: (1) when the manger role reaches the state create role, the involved agent role model framework is triggered. The manager wait till this involved agent role model is validated. (2) An involved agent role model is validated if its responsibilities role and its control role are coherent. (3) A responsibilities role is coherent if it is created by the manager; all its responsibilities appear in the model as states. (4) The control role organizes the role responsibilities and interactions in accordance with their dependencies, etc.

Naturally, the power of this methodology (as of any methodology) resides in its traceability, integrity, and transformability. Traceability is the ability of the supporting environment to track any concept (responsibility, role, state, etc) from one model to another in the integrated methodology. The transformability is the ability of the supporting environment to zoom in/out models, transform a model representation to another (i.e. from state diagram to a programming language). The study of these implementing and validation issues is just being elaborated.

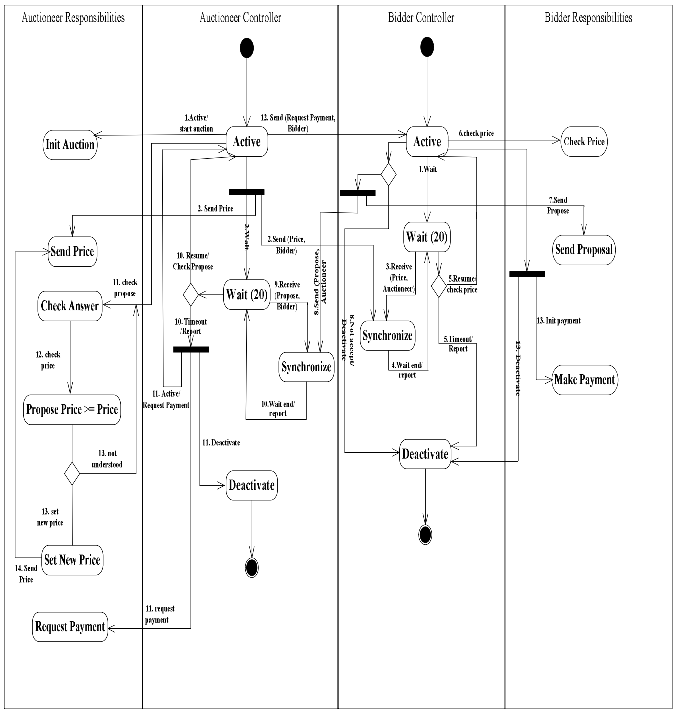

Fig. 15. Responsibilities roles models of English Auction.

Since each role modeling approach has its own characteristics, in order to produce or make a better comparison among them, we will define a set of evaluation criteria:

1) Role integrated modeling methodology, the availability of a methodology guiding and supporting role definition.
2) Manager model, the availability of a model of an organization agent role to add value to the approach, to manage agents' roles in an organization, and to make sure the goals achievement.
3) Control Role Model, the availability of a controller for agent role to manage internal states of agent role.
4) Responsibilities Role Model, The availability of a role model to add value to the approach, to describe roles and their behaviors, can help developers to deal with roles in a well defined way.
5) Supporting tools able to discover and define dependencies and collaboration requirements.

The following table shows a comparison (based on the above criteria) between our proposed methodology and the above studied ones. This comparison states clearly the value of our contribution. In this table, the symbol ● means that there is a full support, while ○ means that the criterion is not supported, and ◘ means that the support is without notation.

This work is at its beginning, and traceability, integrity, and transformability identified problems above and shown in the precedent table are to be solved. Although, no known methodology supports (even partially) these problems, they still remain very important and probably they will be more considered in the future.

TABLE I
COMPARISON BERWEEN METHODOLOGIES

Criteria / Methodology	Integrated methodology	Manager model	Control Role Model	Responsibilities Role Model	Supporting tools
AUML	○	○	○	●	○
Gaia	○	○	◘	◘	○
AMOLA	○	○	●	●	○
MASE	○	○	○	●	○
New methodology	●	●	●	●	○

IV. Conclusion

In this research, we build a coherent methodology for AUML role modeling, to enhance agent role in AUML. So, we can summaries our conclusion in the following points:

1. A methodology for defining agent roles.
2. A model for roles management (defining role, assigning role, un-assigning role and deleting role).
3. A model for agent responsibilities role.
4. A model for roles control (the control of the responsibilities role and its interaction with others roles) in order to achieve a given goal.

References

[1] N. Spanoudakis and P. Moraitis, “The Agent Systems Methodology (ASEME): A Preliminary Report,” In Proceedings of the 5th European Workshop on Multi-Agent Systems (EUMAS'07), Hammamet, Tunisia, 2007.

[2] Y. Zeng and K.. Poh. “Multi-Agent Graphical Decision Models in Medicine,” In Applied Artificial Intelligence: An International Journal, vol.23, pp 103-122, Jan. 2009.

[3] S. Russell, and P. Norvig. *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, NJ: Prentice-Hall, 2002.

[4] N. Spanoudakis, and P. Moraitis, “An Autonomous Agent Application for Product Pricing,” In Proceedings of the 6th European Workshop on Multi-Agent Systems (EUMAS'08), Bath, UK, 2008.

[5] H. Zhu, “Improving Object-Oriented Analysis with Roles”, 6th IEEE International Conference on Cognitive Informatics,pp.430-439,2007

[6] W. Huang, E. El-Darzi, and L Jin (2007), “Extending the Gaia Methodology for the Design and Development of Agent-based Software Systems”, Proceedings of the 31st Annual IEEE International Computer Software and Applications Conference, Peking (Beijing), China, 24-27 July 2007, pp. 159-168.

[7] A. Susi, A. Perini, P. Giorgini, and J. Mylopoulos. “The Tropos Metamodel and its Use.” *Informatica* 29(4), pp. 401-408,2005

[8] L. Ayed, and F. Siala, “From AUML Protocol Diagrams to Event B for the Specification and the Verification of Interaction Protocols in Multi-agent Systems”, Proceedings of the 2008 32nd Annual IEEE International Computer Software and Applications Conference, Turku, Finland, 28 July 2008 -1 Aug 2008, pp.581-584

[9] V.Silva, R.Choren, and C.Lucena, “Using the UML 2.0 Activity Diagram to Model Agent Plans and Actions,” Proceeding of the fourth International Conference on Autonomous Agents and Multi-Agents Systems (AAMAS), Netherlands, 25 July 2005- 29 July 2005. pp. 594-600.

[10] L.Palaniappan, and N.Sambasiva Rao. “Ontology Based Exploiting UML Design of Software Agent Systems (SASE)”. European Journal of Scientific Research, vol.34, pp.358-364, August. 2009.

[11] N. Spanoudakis and P. Moraitis. “The Agent Modeling Language (AMOLA),” Proceedings of the 13th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA 2008), Varna, Bulgaria,2008, pp. 32-44.

[12] L.Padgham, J.Thangarajah, and M.Winikoff, “AUML protocols and code generation in the Prometheus design tool,” Sixth International Joint Conference on Autonomous Agents and Multi-agent Systems, Hawaii, USA, 2007.

[13] M.P. Huget and J. Odell. “Representing Agent Interaction Protocols with Agent UML’’. In AAMAS’04, 2004, pp.1244-1245.

[14] I. Trencansky and R. Cervenka. “Agent Modeling Language (AML): A Comprehensive Approach to Modeling MAS.’’ Informatica, vol. 29, No. 4, pp. 391-400, 2005.

[15] M. Huget, J. Odell, and B. Bauer, “The AUML Approach, in Methodology and Software Engineering for Agent Systems,’’ eds. Bergenti, F., Gleizes, M.-P., Zambonelli, F., Kluwer Academic Publishers, 2004, pp. 237-257.

[16] M. Huget, “Agent UML Class Diagrams Revisited,’’ In Proceedings of Agent Technology and Software Engineering (AgeS), Bernhard Bauer, Klaus Fischer, Joerg Mueller and Bernhard Rumpe (eds.), Erfurt, Allemagne, October 2002.

[17] Q.Wang, Y.Zhang, and C.Xing. “Object Oriented Design Based on Role and Rule.’’ compsac, 2008 32nd Annual IEEE International Computer Software and Applications Conference, 2008, pp.779-784.

[18] J.Leng, J.Li, and L.Jain. “A Role-based Framwork for Multi-agent Teaming.’’ 12th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems,2008, pp.642-649

[19] S. Kirn, O. Herzog, P. Lockemann, and O. Spaniol. Multi-agent Engineering: Theory and Applications in Enterprises. Berlin: Springer, 2006.

[20] H Xu, X Zhang, and R. J. Patel. “Developing Role-Based Open Multi-Agent Software Systems.’’ *International Journal of Computational Intelligence Theory and Practice,* June 2007, Vol. 2, No. 1, pp. 39-56.

[21] Marc. Huget. “Generating Code for Agent UML Sequence Diagrams,’’ In Proceedings of Agent Technology and Software Engineering (AgeS), Bernhard Bauer, Klaus Fischer, Joerg Mueller and Bernhard Rumpe (eds.), Erfurt, Allemagne, October 2002.

[22] M.Cossentino. “From Requirements to Code with the PASSI Methodology,” in *Agent-Oriented Methodologies*. Hershey, PA, USA: Idea Group Publishing, 2005, pp.79–106.

[23] I. Nunes, U. Kulesza, C. Nunes, and C. de Lucena. “Extending PASSI to Model Multi-agent Systems Product Lines.” ACM Symposium on Applied Computing - SAC, 2009, pp. 729-730.

[24] J. Ferber, O. Gutknecht, and F. Michel. “From Agents to Organizations: An Organizational View of Multi-agent Systems.” in Agent-Oriented Software Engineering IV, 2004, pp. 443-459.

[25] G. Booch J. Rumbaugh and I. Jacobson. *The Unified Modeling Language User Guide*. USA: Addison-Wesley Professional, 2005, pp.123-35.

[26] B. Henderson-Sellers and P. Giorgini. *Agent-Oriented Methodologies*. in Hershey, PA, USA: Idea Group Publishing, 2005.

[27] N. Spanoudakis and P.Moraitis. “An Agent Modeling Language Implementing Protocols through Capabilities,” In: Proceedings of The 2008 IEEE/WIC/ACM Int. Conference on Intelligent Agent Technology (IAT-08) , Sydney, Australia, 2008.