HoloJade:
A Role Based Holonic Extension for JADE

Ahmad Esmaeili, Iran University of Science and Technology, Iran

Nasser Mozayani, Iran University of Science and Technology, Iran

ABSTRACT

Holonic Multi-Agent Systems (HMAS) provide a convenient and relevant way to analyze, model, and simulate complex systems in which a large number of entities are interacting at different levels of abstraction. Many models have been proposed for the implementation of these systems; however, most are not general enough to cover applications other than the ones for which they are applied. In this paper, the authors introduce HoloJade, an extension to JADE platform, as a generic solution for the development of HMASs in which Holons and their assigned roles are presented as first level entities available at runtime. This includes a detailed description of the extension, in terms of its meta-model, the needed protocols for the possible interactions, and facilities for the reorganization of the holons. In this paper, the authors also present a hypothetical library example to demonstrate the steps for designing a holonic structure using this extension.

Keywords: *HMAS, HoloJade, Holon, JADE, Multi-Agent Systems*

INTRODUCTION

During recent decades there has been an immense growth in size and complexity of Multi-Agent Systems (MASs) (Odell et al., 2005). Although MASs are considered today to be well suited to analyze, model and simulate complex systems, in cases where there is a great number of entities interacting at different levels of abstraction, it seems improbable that MAS will be able to faithfully represent complex systems without multiple granularities. To deal with this problem, holonic systems have attracted the attention of researchers. And today many of its contributions to many application ranging from Manufacturing Systems (Maturana et al., 1999), Transports (Burkert et al., 1998), cooperative work(Adam et al., 2000) or yet radio mobile mesh dimensioning (Rodriguez et al., 2003) are apparent.

Holonic Structures like Organizational ones in multi-agent systems are beneficiary of many desirable and simplifying features such as simplification in representation, further encapsulation, and modularization. In these structures, interactions are specified in two different scopes, within and outside of the holon, and as a result interoperability problems are reduced by a considerable degree. Additionally, by means of Role concept and evaluation of the holons or organizations and their members using this concept, we can reduce the possibility that unfit members can be grouped together toward

DOI: 10.4018/jats.2010040104

a specified aim and on the other hand, enable them comply with these requirements. In spite of over-mentioned useful features, these kinds of structures are not always supported explicitly and generically by agent platforms and most of the proposed solutions or platforms are application based, i.e., they have been developed for a specific domain of applications.

Considering these remarkable features of holonic systems in solving and modeling of complex problems, it is not surprising to find numerous methodologies and frameworks for the development of these systems. Most of the works proposed in implementation and development of these systems, as stated above, are strongly related to the domain of applications, mostly in manufacturing domain, they were applied for. In other words, an agent or holonic architecture to that specific application is presented at first and then that architecture is implemented using current agent platforms as desired. Therefore these solutions are applicable merely to very similar cases. For instance, Van Brussel et al. (1998) introduced a holonic system architecture, called PROSA that includes three types of basic holons: order holons, product holons and resource holons; Langer proposed a methodology and architecture for holonic multi-cell control system (Langer, 1999); Liu et al. (2000) presented the architecture and coordination of a holonic automated guided vehicle system, and Glanzer et al. (2001) implemented a machine-holon using ZEUS (Nwana et al., 1999) agent framework.

In addition to the over-mentioned examples, some of other works try to propose a generic framework that is applicable in various domains. Among these works, we can mention the platform developed by Rodriguez et al. (2005). In their work, the authors have used MADKIT (Gutknecht & Ferber, 2000a; Gutknecht & Ferber, 2000b) agent platform which is built upon the AGR (Agent / Group / Role) organizational model (Ferber et al., 2004; Gutknecht, 2001) and extended it to support holonic structures by considering Parunak and Odell's modification to AGR model. Another generic framework which is worth noting is JANUS (Gaud et al., 2008). This platform has been developed from scratch to support holonic multi-agent systems based on an organizational approach and its key focus is that it supports the implementation of the concepts of role and organization as first-class entities.

In this paper we present HoloJade, a holonic extension for JADE platform (Bellifemine et al., 1999; JADE, 2001). JADE platform is a FIPA (2002) compliant software framework for multi-agent systems in Java that allows the coordination of multiple agents and the use of directories and the standard FIPA-ACL communication language in both SL and XML. The agent platform can be distributed across machines (even with different OS) and its configuration can be changed at run-time by moving agents from one machine to another whenever it is required. JADE supports the implementation of ontology for the content of messages and knowledge of agents and since it provides a library of behaviours for performing FIPA interaction protocols, is also one of the preferred platforms to implement conversation protocols among agents either from scratch or by combining the existing protocols. In spite of addressing the problem of composition of agent groups, it does not provide explicit features for groups apart of the emergent behaviour obtained by manifesting the behaviours of each agent. It is also does not support the role concept. In this paper we try to add the concepts of holon ad roles as first class entities to this popular platform by the HoloJade extension.

In the rest of this paper, in section two an introduction to holonic systems is given. In section three, our HoloJade extension is presented by its detailed meta-model and required protocols. In section four, we will show how this example can be used for the modeling of a hypothetical holonic library example. Finally in section five conclusion and remarks for future works are presented.

Holonic Multi-Agent Systems

The concept of holon is central to this paper and therefore a definition of it seems to be

helpful before we continue. The term holon was introduced for the first time by the philosopher Arthur Koestler (1967) in order to explain the evolution of biological and social systems. He made two key observations:

- These systems evolve and grow to satisfy increasingly complex and changing needs by creating stable "intermediate" forms which are self-reliant and more capable than the initial systems.
- In living and organizational systems it is generally difficult to distinguish between 'wholes' and 'parts': almost every distinguishable element is simultaneously a whole (an essentially autonomous body) and a part (an integrated section of a larger, more capable body).

In multi-agent systems, the vision of holons is much closer to that of Recursive or Composed agents among the agent community. A holon constitutes a way to gather local and global, individual and collective points of view. Therefore, holon is a self-similar structure composed of other holons as sub-structures. This hierarchical structure composed of holons is called a *holarchy*. Depending on the level of observation, a holon can be seen either as an autonomous *atomic* entity, or as an organization of holons. In other words a holon is a whole-part construct that is composed of other holons, but it is, at the same time, a component of a higher level holon.

In a holonic multi-agent system, we can distinguish between two main types of holons, namely *head* and *body* holons. All of the holons which are a member of another holon (called *super holon*), are considered to be a body holon. These holons can be either atomic or composite and are performing the tasks that have been delegated to them. Holons of type head act as representatives of the holons they are members of. In other words, holons are observable to the outside by communication with these representatives. As the representatives, head holons manage the holons communication with the outside of the holon in pursuit of the goal of the holon and coordinates the body holons in pursuit of these goals. In a holon, the force that keeps the heads and bodies is their commitments to the goal of the holon. It is worth noting that in this commitment the relationships among the agents and holons are formed at runtime, in contrary to classical methods such as object-oriented programming in which they are expressed at code level. More precisely according to (Schillo & Fischer, 2003), for a MAS with the set A of agents at time t, the set H of all holons is defined recursively as:

- every instantiated agent can be considered as an atomic holon.
- h = (Head, Sub-holons,C) $\in$H, where Sub-holons $\in 2H \setminus \emptyset$ is the set of holons that participate in h, Head $\subseteq$ Sub-holons is the non-empty set of holons that has the responsibilities as defined above, and C $\subseteq$ Commitments defines the relationship inside the holon and is agreed on by all holons h´ $\in$ Sub-holons at the time of joining the holon h and keeps them inside the holon.

According to the definitions above, a holon h is observed by outside world of h like any other agent in A. Only at closer inspection it may turn out that h is constructed from (or represents) a set of agents. Like traditional agents any holon has a unique identification, this makes it possible to communicate with each holon by just sending messages to their addresses and the heads/ head of the holon is the one that handles these messages. Given the holon $h = (Head, \{h_1, ..., h_n\}, C)$ we call $h_1, ..., h_n$ the *sub-holons* of h, and h the *super-holon* of $h_1, ..., h_n$. The set $Body = \{h_1, ..., h_n\} \setminus Head$ (the complement of *Head*) is the set of *sub-holons* that are not allowed to represent holon h. Holons h' are allowed to engage in several different holons at the same time, as long as this does not contradict the sets of commitments of these *super-holons*.

Holons and Roles in HoloJade

In this paper we have used the definition of *Agentified Group* in (Odell et al., 2005) to define Holons in JADE. In other words, a holon, atomic or composite, possesses all of the features an agent might have such as sending/receiving messages directly and taking on roles as the specifications of its behaviours. Holonic structures can be established at design time by the system designer or formed dynamically at run time according to the collective goals or tasks that may arise. Furthermore in order to perform the common tasks of the holon, they should be capable of distributing responsibilities in form of tasks to its members according to the roles the can play. In HoloJade we have concentrated on Holons and Roles as first-class entities, Holon as a computational entity of the framework and Role as a description of the requirements the members should fulfill.

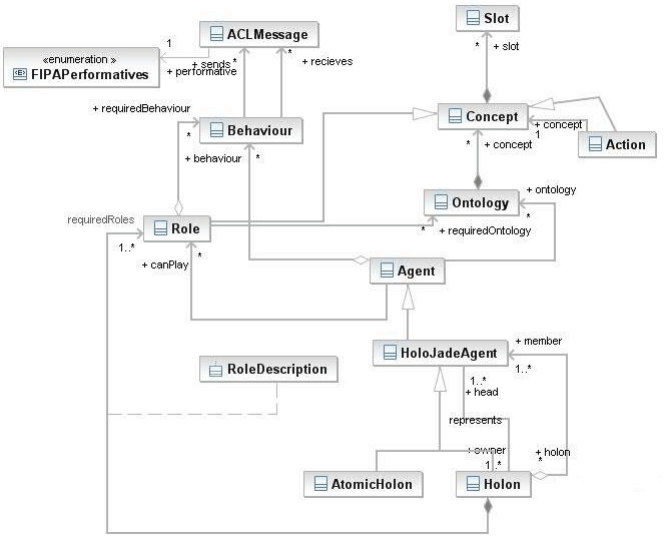
In the rest of this section we propose a meta-model for HoloJade to support these features. For this purpose, at first step, the class model of JADE has been extracted from JADE's source code and documentation. Adding the extensions, the HoloJade model is serialized again into Java source code. During these processes, in order to take advantage of the transformation tools available in eclipse, we have made use of Eclipse Modeling Framework (EMF) (Budinsky et al., 2003) and the model is implemented in Eclipse Modeling Framework Ecore Language.

The HoloJade Meta-Model

HoloJade extension mainly focuses on the concepts / classes of *HoloJadeAgent*, *Holon*, and *Role*. The partial view of the core of HoloJade meta-model can be seen in Figure 1.

As it has been depicted, Instead of using the Agent class of JADE directly, we have specialized the Agent class in HoloJadeAgent class. This class not only inherits all of the features and capabilities of the Agent class, but also implements and contains the needed data structures and methods to manage holonic multi-agent systems such as membership information of the holons and the roles they are capable of playing. In HoloJade we have separated the representation of the atomic and composite holons using AtomicHolon and Holon classes respectively. This separation in representations of these two types of holons is

Figure 1. Partial view of HoloJade metamodel



because of the fact that some information kept about the composite holons are not applicable to the atomic ones, and representing these two holons both by a single entity will cause some vagueness in their management.

As shown in Figure 1, the Holon class is engaged in two bidirectional relationships with the HoloJadeAgent class. By means of the aggregation the Holon class holds references to all of its members which can be either atomic or composite ones. As mentioned, this relation is bidirectional and that causes a member holon to contain its membership info about the holons it is a member of. In addition to this aggregation relationship, there is also a bidirectional association relationship to specify the representatives (heads) of the holon. Again, since this relationship is defined bi-directionally, the heads also keep references to the holons they are representing or managing. According to the fact that in a holonic system a composite holon might be considered as an atomic one depending on the level of abstraction, in the HoloJade model as mentioned before, the HoloJadeAgent class in general and the Holon class in particular extends the JADE's Agent class. This causes an instance of the Holon class to possess its own set of behaviours, to be able to perform tasks as a whole, and to interact and communicate with the other entities directly.

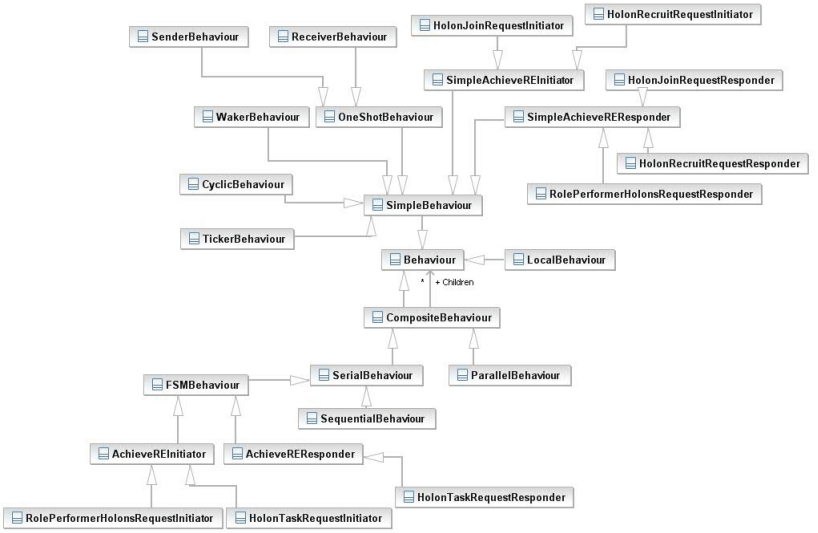
One of the other key components in HoloJade is the concept of Roles. As it was stated before, this concept is employed to have a better control over the capabilities of the holons in general and simply and effectively define the interaction between the holons, their requirements and constraints. In Figure1 the use of roles in HoloJade is depicted by means of two association relationships, *canPlay* and *requiredRoles*, between the *Role* class, the *HoloJadeAgent* and the *Holon* classes respectively. It should be noted that in defining the required roles for a holon, sometimes we need to consider extra restrictions or requirements about the memberships made by playing a specific role inside a holon. In HoloJade's meta-model these restriction can be specified using the association class RoleDescription, as shown in Figure1. In the HoloJade, the Role class is implemented as an Ontology Concept, part of the *HolonOntology*. We might think of different properties for the Role class. Among these properties, required behaviours, required ontologies, and message templates for the sent and received messages seem to be the most important ones. Although these properties altogether provide many criteria and requirements to be checked, only some of them might be actually specified depending on the evaluation strategy used by the system designer. For example, the required ontologies allow evaluation of the knowledge available to the agent and similarly the list of required behaviours can potentially be used either in order to check whether the Holon is able to perform the Role's tasks or in order to allow the Holon to acquire the behaviours required to fulfill the role, by adding them to its known behaviours depending on the value of the Role's properties.

In JADE framework, the complex behaviour hierarchy allows the different ways for the execution and nesting of them. It also provides an implementation for various FIPA protocols together with the required classes for the implementation all the other possible protocols defined by FIPA. For instance, the *SimpleAchieveREInitiator* and *SimpleAchieveREResponder* are provided to implement all the FIPA-Request-like interaction protocols in which an initiator sends a single message to the Participant in order to verify whether the RE (Rational Effect) of the communicative act has been achieved or not. The *Behaviour* class shown in Figure1 is only the root of the actual hierarchy mentioned above. We have tried to present a more complete but again partial view of the Behaviour hierarchy after applying the needed extensions, in Figure 2.

HoloJade Protocols and Interactions

In HoloJade, the holonic structures are able to be established either at design time or run time. As for the reorganization of the holonic structures at run time, the extension is provided with proper functionalities of registration/

Figure 2. Partial view of the Behaviour hierarchy in HoloJade



deregistration, recruiting, and joining of new holons. These tasks are performed using communication protocols which are described in detail further on.

Publishing to Directory Facilitator

Establishment of the holonic structures at design time can be carried out by the initialization of them as discussed before; however, for those determined at run time, a proper mechanism is needed to choose among the holons according to the roles they play or require. JADE already provides a directory service called the Directory Facilitator (DF) through which an agent can look for other agents with specific set of features such as the protocols supported or the ontologies it can access. Details on defining DF services and querying them can be found in (Bellifemine et al., 2007).

In HoloJage, we have extended *DFAgentDescription*, which is a part of JADE's *FIPAAgentManagementOntology*, in order to keep track of the descriptions of holons, considering their structures and capabilities (Figure 3). As it can be seen in this figure, *DFAgentDescription* is extended in to *DFHolonicMemberDescription* in order to provide the descriptor of the holonic members. In this hierarchy there is also an association relationship between *DFHolonicMemberDescription* and the *Holon* class. This reference to the super-holons of a holon, allows it to communicate with the other holons following the rule according to which, holons are permitted to merely have a direct communication with the holons in the same super-holon (intra-holonic communication); while the inter-holonic communications are provided through the heads of the holons. Now by means of this relationship a holon can simply recognize and look for the other holons inside its supper-holon.

Since the Holon requires roles (as a whole) and plays them (as a part), the Holonic descriptor, *DFHolonDescription*, was created by extending *DFHolonicMemberDescription* with a list of required roles (Figure 3).

Reorganization of the Holonic Structures

In order to let the holonic structures to be established at runtime, the description of all of the holons and their members are registered in the DF. Then, in the process of establishment, as the first step, a search for a proper holon is carried out by querying the DF service. Once the list of candidate *DFHolonicMemberDescriptions* or *DFHolonDescription* is retrived, the holon initiates *HolonRecruitRequest* protocol (in case that a holon tries to recruit a new member) with the holon it wants to join (Figure 4). This protocol is a simplified version of the FIPA Contract Net Interaction Protocol (2002) that has been implemented using *HolonRecruitRequestInitiator* and *HolonRecruitRequestResponder behaviours* (Figure 2). Before proceeding on the discussion over the protocols, it should be noted that, the head of the holon is responsible for all of decision making processes. Therefore whenever we use the word holon as the requester or participant, that is the head of the holon (in case it is composite) that actually handles the communication/interaction and makes decisions.

In recruiting process, the retrieved candidate holons and the holon which wants to employ a new member take the *Participant* role and the *Requester* role respectively. The Requester holon sends a Call For Proposal (CFP) with a role object as the content, to the Participants. Once this CFP is received by the Participant, an ACL refuse message is produced if the request is denied because of failure in decision process discussed later. In case that the Participant does not have any problem, it

Figure 3. Directory description class hierarchy of Holon Ontology

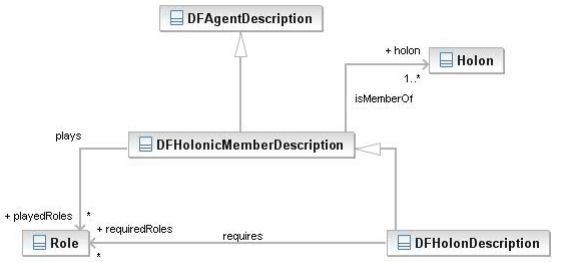
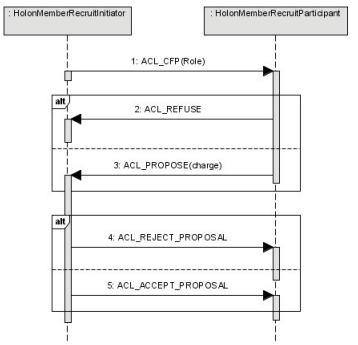


Figure 4. HolonRecruitRequest Protocol



sends an ACL propose message with the calculated charge of joining and performing the requested role, as the content of message. Now the Requester can either: (i) reject the proposal and the interaction terminates or (ii) accept the proposal and inform the Participant. Now, being informed, the Participant joins as a member to the Requester holon. As it can be expected, the decision process for refusing, rejecting, or accepting these requests is left to other internal behaviours of the holon. A similar and even simpler protocol can be applied for the holon that wants to join another holon (HolonJoinRequest). We depict this protocol without any further discussion in Figure 5.

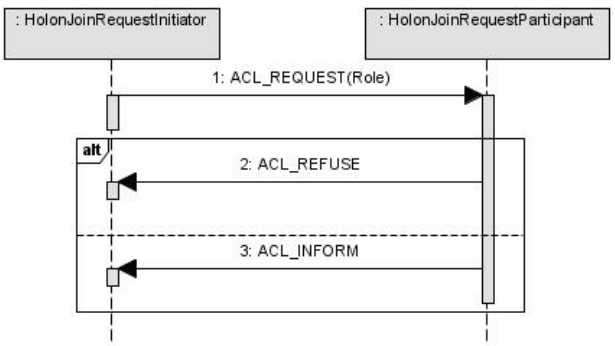
In the decision process of every single step of the protocols discussed above, various criteria might be considered to verify. The complete list for these criteria may be recognized at design time according to the application that this framework is used for. However we present some of the trivial ones here which can be applied in many cases. These evaluation options are based on the concept of the Role and are defined as follows:

- **Eligibility:** This option can be checked using the canPlay association between the HoloJadeAgent and the Role (Figure 1) through the type/class definition.
- **Required Behaviours:** The list of required behaviours lets us verify whether the prospective holon "knows" the behaviours that implements the desired protocols.
- **Message Templates:** A basic list of messages that should be sent or received by the holons as part of the interactions/protocols can be specified using the message template mechanism provided by JADE.

Holon's Task Distribution

For the purpose of letting the members of a holon manage their tasks and work load, we present a simplified version of the FIPA Request Protocol (2002), as *HolonTaskRequest* protocol (Figure 6) which is implemented using the *HolonTaskRequestInitiator* and *HolonTaskReqestResponder* behaviours (Figure 2). Here the requester holon chooses among the candidate members the best one that fits the task, using a mechanism implemented as part of the HolonTaskRequestInitiator behaviour. Then it sends an ACL request message to the chosen holon as the Participant. Upon receiving the request message, the Participant might refuse it because of being busy, or accept to perform the task by sending an ACL accept message to the requester. Finally, the Participant holon sends an ACL inform/failure message to the requester in case it successfully performs the task or founds a problem during the execution of it, respectively.

Figure 5. HolonJoinRequest Protocol



Case Study: A Library System

In most of the works dealing with the multi-agent systems and frameworks, the traditional example of *Buyer* and *Seller* agents are used as a case study. However, this example does not fully exhibit the various parts of our framework. For this purpose, in this section we give a hypothetical library example to show how the holonic structures are set up by means of HoloJade. In this example as depicted in Figure 7, we have seven main holons that are grouped as follows: (i) the composite holon, *University*, which contains two atomic holons, namely *ComputerStudent* and *ComputerProfessor*, and the composite holon *Library* with the atomic holon *Librarian* as its member, (ii) the composite holon, *BookStore*, with the atomic holon, *Salesman*, as its member. In order to keep the interactions among the holons in this system as simple as possible, we will use some simplifying assumptions that will be described wherever they are applied.

In this system, the scenario of borrowing a book is as follows: when a *ComputerStudent* tries to borrow a book, if the *Library* owns the book (available or already checked out), he receives the book or a proper response about the status of the book. In the case that the book is necessary for the *ComputerStudent* and there is no such a book in the *Library*, according to its benevolent system and by providing a formal letter from a *ComputerProfessor* to confirm his need, the *Student* can have the *Library* purchase the book from a *BookStore* and put it in the library.

Before proceeding to the detailed process using the concepts we defined in HoloJade, we present the holonic structure of the system together with the necessary roles (but without the presenting the heads for clarity) in it as Figure 8.

According to this structure the detailed scenario is as follows: the ComputerStudent holon find the address of the Library holon by searching in the DF. Then it sends a request to the Library to join it by playing the *Borrower* role (Using the previously defined HolonJoinRequest Protocol). Having been accepted and joined, the ComputerStudent sends the information of the book he wants to borrow to the Librarian holon. The Librarian processes the in-

Figure 6. HolonTaskRequest protocol

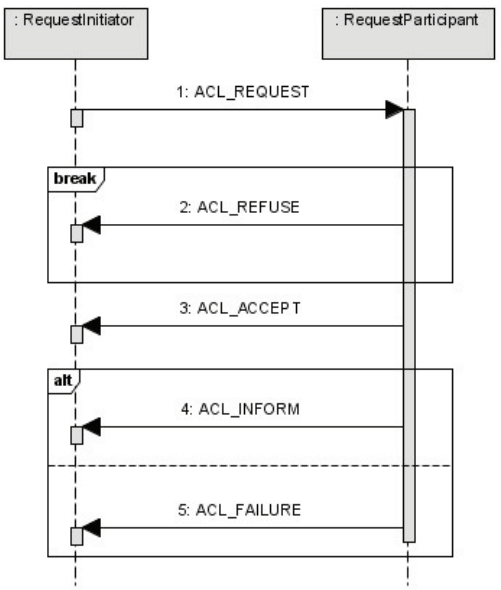


Figure 7. Holons in the library system

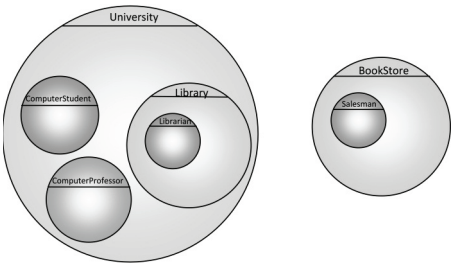
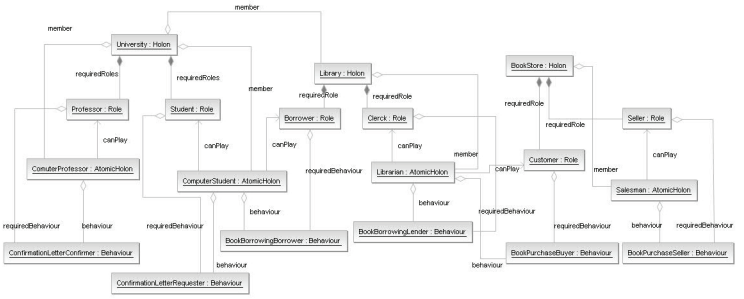


Figure 8. Holonic structure of the library system



formation and then three cases might occur: (i) the book is available and is lent to the ComputerStudent, or (ii) the book has been checked out and the necessary information is given to the ComputerStudent, or (iii) there is no such a book in the Library. These interactions are illustrated in Figure 9.

Being aware of the generous service provided by the system and really in need for the book, the ComputerStudent obtains a confirmation letter from the ComputerProfessor holon (address of which is obtained through the DF). This can be carried out using a very simple version of the FIPA Request Protocol (2002) and is not explained further here. Now the ComputerStudent gives the letter to the Librarian in order to be verified. In the case that there is no problem with the letter, the Librarian tries to purchase the book. In this process, the Librarian first obtains the address of the BookStore holon (in which the book is available) using DF and through the permission of heads of its upper level holons. Then it commences an interaction with the BookStore Library for playing the *Customer* role, again using the HolonJoinRequest protocol. Having joined the BookStore, played the Customer role to purchase the book, and registered it in the library system, the Librarian lends it to the ComputerStudent (there is no need to decide about the acceptance of the price of the book since it is almost constant in all of the bookstores). This scenario is depicted in Figure 10.

It is important to note that, For the purpose of simplifying the interactions and communications, we did not present the interactions with the heads of holons as part of the processes specified before. However, during its implementation process using the HoloJade, all of the inter-holon communications and decision makings are taken place through or by the permission of the head holons.

Figure 9. Book borrowing protocol

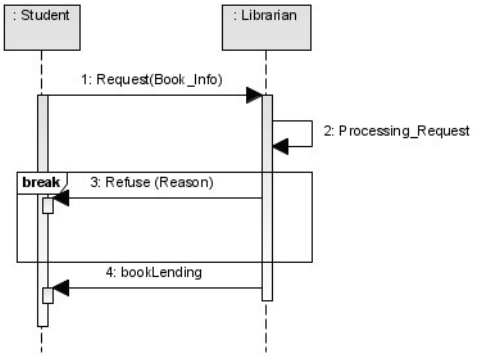
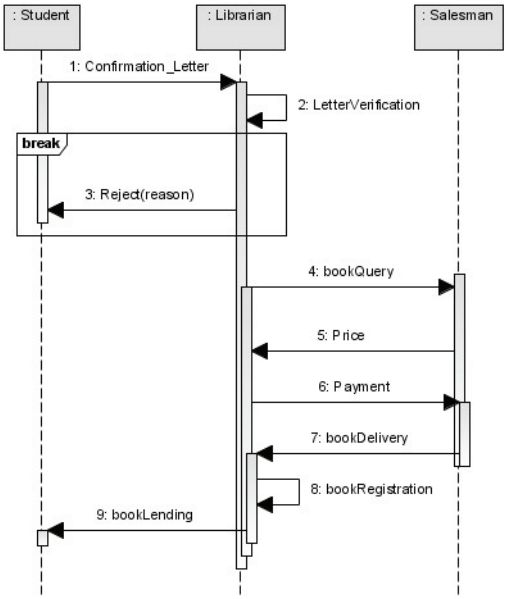


Figure 10. Book preparing protocol



We are aware that this is a very small and simple case study of a very trivial situation. However, we consider that it does reveal how the use of holons and roles can help encapsulate and group functionalities and interactions in a clear and intuitive way, while still allowing some flexibility in the composition of the holons at run time.

CONCLUSION

Holonic multi-agent systems are very powerful tools for dealing with large and complex problems. In spite of its ever-increasing usage in solving real world problems, especially in manufacturing systems, there are a few numbers of FIPA-compliant platforms that support the

development of them, and are not dependant on a specific domain of application. On the other hand JADE is a very powerful multi-agent development framework that is completely FIPA-compliant and supports the implementation of a wide range of features in both scientific and commercial applications. In this paper we made an attempt to bring the concept of holons and roles into the JADE as an extension to enables it in the development of these structures. Through the article, we tried to present this extension by defining the necessary class and concepts together with the required basic protocols for the possible interactions among the holons. In this work, for the better understanding of the process, we presented a small holonic example with a detailed explanation of its implementation using the HoloJade extension. This shows how this extension can be used in a similar way to develop holonic structures.

We believe that this extension can be improved further to provide better results. For instance, since there is no centralized protocol description in JADE, and only a projection of each role is implemented in a behavior, it is sometimes hard to guarantee that any accepted candidate holon can properly fit a given role. Although, we have provided the three trivial evaluation methods: eligibility, required behaviours and message templates, a better mechanism for this evaluation is indeed a very important factor in the performance and accuracy of HoloJade. We recommend this as a future work.

REFERENCES

Abstract Architecture Specification, F. I. P. A. (2002). *Foundation for Intelligent Physical Agents: FIPA Abstract Architecture Specification* (Document No. SC00001L). Retrieved August 2009, from http://www.fipa.org/specs/fipa00001/SC00001L.html

Adam, E., Mandiau, R., & Kolski Homascow, C. (2000). A holonic multi-agent system for cooperative work. In *Proceedings of the 11th International Workshop on Database and Expert Systems Applications* (pp. 247-253).

Bellifemine, F., Caire, G., Trucco, T., & Rimassa, G. (2007). *Jade Programmer's Guide*. Retrieved from http://jade.tilab.com/doc/programmersguide.pdf

Bellifemine, F., Poggi, A., & Rimassa, G. (1999). JADE–a FIPA-compliant agent framework. In Proceedings of Practical Applications of Intelligent Agents (pp. 97-108).

Budinsky, F., Steinberg, D., Merks, E., Ellersick, R., & Grose, T. (2003). *Eclipse Modeling Framework*. Reading, MA: Addison-Wesley.

Burckert, H. J., Fischer, K., & Vierke, G. (1998). Transportation scheduling with holonic mas – the teletruck approach. In *Proceedings of the 3rd International Conference on Practical Applications of Intelligent Agents and Multiagents* (pp. 577-590).

Contract Net Interaction Protocol, F. I. P. A. (2002). *Foundation for Intelligent Physical Agents: FIPA Abstract Architecture Specification* (Document No. SC00001L). Retrieved August 2009, from http://www.fipa.org/specs/fipa00029/SC00029H.html

Ferber, J., Gutknecht, O., & Michel, F. (2004). From agents to organizations: an organizational view of multi-agent systems. In *Proceedings of the 4th International Workshop of Agent-Oriented Software Engineering IV (AOSE)* (LNCS, pp. 214-230).

Gaud, N., Galland, S., Hilaire, V., & Koukam, A. (2008). An organisational platform for holonic and multiagent systems. In *Proceedings of the Workshop on Programming Multi-Agent Systems*, Estoril, Portugal.

Glanzer, K., Hammerle, A., & Geurts, R. (2001). An Integral Implementation of a Machine-Holon Applying the ZEUS Agent Framework. In *Proceedings of the Multi-Agent-Systems and Applications II (HoloMAS, 2001)* (pp. 296-307).

Gutknecht, O. (2001). *Proposition dun Modèle Organisationnel générique de système multi-agent. Examen de ses conséquences formelles, implémentatoires et méthodologiques. Unpublished doctoral disseration*. Montpellier, France: Université de Montpellier II.

Gutknecht, O., & Ferber, J. (2000a). Madkit: a generic multi-agent platform. In *Proceedings of the 4th International Conf on Autonomous agents* (pp. 78-79). New York: ACM Press.

Gutknecht, O., & Ferber, J. (2000b). The MADKIT agent platform architecture. In *Proceedings of the Agents Workshop on Infrastructure for Multi-Agent Systems* (pp. 48-55).

JADE. *Java Agent Development Framework*. (2001). Retrieved August 2009, from http://jade.tilab.com

Koestler, A. (1967). *The Ghost in The Machine*. Hutchinson, KS: Hutchinson.

Langer, G. (1999). *A methodology and architecture for holonic multi-cell control system*. Unpublished doctoral dissertation, Technical University of Denmark, Lyngby, Denmark.

Liu, S., Gruver, W. A., Bardi, S., & Kotak, D. (2000). Holonic manufacturing system for distributed control of automatic guided vehicles. In *Proceedings of IEEE International Conference on System, Man and Cybernetics,* Nashville, TN (pp. 1727-1732).

Maturana, F., Shen, W., & Norrie, D. H. (1999). MetaMorph, an adaptive agent-based architecture for intelligent manufacturing. *International Journal of Production Research*, *37*, 2159–2174. doi:10.1080/002075499190699

Nwana, H., Ndumu, D., Lee, L., & Collis, J. (1999). ZEUS: a tool-kit for building distributed multi-agent systems. *Applied Artificial Intelligence Journal*, *13*(1), 129–186. doi:10.1080/088395199117513

Odell, J., Nodine, M., & Levy, R. (2005). A Metamodel for agents, roles, and groups. In *Proceedings of the 5th International Conference on Agent-Oriented Software Engineering (AOSE)* (LNCS, pp. 78-92).

Request Protocol, F. I. P. A. (2002). *Foundation for Intelligent Physical Agents: FIPA Request Interaction Protocol Specification (2002)* (Document No. SC00026H). Retrieved from http://www.fipa.org/specs/fipa00026/SC00026H.html

Rodriguez, S., Hilaire, V., & Koukam, A. (2003). Towards a methodological framework for holonic multi-agent systems. In *Proceedings of the 4th International Workshop of Engineering Societies in the Agents World*, Imperial College, London (pp. 179-185).

Rodriguez, S. A. (2005). *From analysis to design of holonic multi-agent systems: A framework, methodological guidelines and applications*. Unpublished doctoral dissertation, Université de Technologie de Belfort-Montbéliard, Belfort, France.

Schillo, M., & Fischer, K. (2003). Holonic multiagent systems. In *Proceedings of KI, 2003* (Vol. 17, pp. 54-55).

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., & Peeters, P. (1998). Reference architecture for holonic manufacturing systems: PROSA. *Computers in Industry*, *37*, 255–274. doi:10.1016/S0166-3615(98)00102-X

Ahmad Esmaeili received his B.Sc. degree in Computer Engineering from Urmia University in 2007, and now he is pursuing M.Sc. in Artificial Intelligence at Iran University of Science and Technology (IUST). His research interests include Multi-Agent Systems, Evolutionary Computations, and Data Mining. He is currently working on his M.Sc. Thesis in which he intends to use Holonic Multi-Agent Systems in traffic management.

Nasser Mozayani received a B.Sc. degree in Electrical Engineering (Computer Hardware) from Sharif University of Technology (Tehran, IRAN); M.Sc. degree in Information Systems from Supelec (Rennes, France) and a Ph.D. degree in Informatics from University of Rennes 1 (Rennes, France). His research interests are: Soft Computing and Multi-Agent Systems. He is currently assistant professor in the Department of Computer Engineering at Iran University of Science & Technology.