Towards an extension of UML2.0 to model mobile agent-based systems

Mohamed Redha Bahri, Rabah Mokhtari, and Allaoua Chaoui

MISC Laboratory, Department of Computer Science, University Mentouri Constantine
Constantine, 25000, Algeria

Summary
The technology of mobile agents obtained recently more importance not only because of its capacity of developing and building a distributed, heterogeneous, and interoperable systems, but also because of its robustness development of mobile and communication network as well. However, there are few works dealing with the methods and tools of analysis and design of the mobile agents systems. Furthermore, the mobile systems have introduced new concepts as: ***migration***, ***cloning*** and the ***locations***. We propose in this paper an extension of the most important UML 2.0 diagrams to model the mobile agents systems with the objective to face these three concepts. A case study illustrates the proposed approach.
Key words:
Mobile agents, Object oriented agent, UML, object oriented modeling approach.

1. Introduction

Mobility concerns in general the physical movement of device material as well as the migration of software applications. Indeed, many concepts emerged in the field of mobility. As examples, we may cite: first, the mobile code which is defined by the capacity of changes dynamically, the attributions between the code's fragments to be executed and their execution locations. Second, the mobile calculus, which is a paradigm in which a device mobile users got access to the infrastructures that are divided independently from their physical locations. Finally, the mobile agents which are software entities are able to move from a network site to another, in an autonomous way to reach their goals. Though, the mobile agents were recently introduced, they gave birth to two great standardization norms (FIPA [2] and MASSIF [6]) as well as an important number of development platforms [6]. However, the designers and developers of the mobile agents applications are banged against constrains during the analysis and conception phases.
In order to engender the mobile agents Applications the predominant approaches oriented agent that exist actually, as the AAII approach of kinny and al [5] inspired form Rumbaugh work and UML agent of Bauer and al [1] based on UML extension, tried to adapt the object oriented analysis and design methodology [11] a long with the modeling language UML [10]. But, these approaches are limited to the oriented agent which is unable to capture the mobility concepts characterizing the mobile agents. On the other hand, it is advantageous to get a useful approach throughout the process development of the mobile agents, getting the inspiration from the notes of the object oriented analysis and design (in particular from UML language) .However, the relation-ship between the agents in general and the objects remains an open problem [11].

Our contribution is based on UML2.0, because it provides us with new modeling elements, more flexible than their predecessors and better prepared to capture the mobility concepts. The multidimensional hierarchical partitions, or else, the multi-hierarchic swimlanes are examples in which one dimension represents a location and the other one represents an object. This can be easily adapted to our proposal in both: statechart and activity diagrams.
The following part of this paper is organized as follows: section 2 deals with similar works that we consider interesting to cite. Section 3 exposes our contribution that consists in the extension of different diagrams of UML 2.0 to support the mobility. Section 4 will illustrate the approach with a case study of electronic award system and will conclude with a general conclusion and perspectives.

2. Related Work

UML is the language of modeling, adapted and standardized by the international community oriented object. However, UML suffers from a lack of rigorous formalization to test the semantics of the elaborated models. In this context, many works were realized, some of them were based on the CCS (calculus of concurrent process) [7] with the objective to present analysis models of mobile systems in general and mobile agents in particular. Others proposed modeling formal approaches as Dianxiang XU and al [12] who proposes to model the mobile agent-based systems with higher level Petri-network. In the works of k.Saleh and al [9], they tried to extend UML by mobility, and proposed M-UML as a complete extension of all the UML 1.4 standard diagrams. We may also cite the works of M.Kang and al [3] through which, the authors present an approach to model the

Manuscript received October 5, 2009
Manuscript revised October 20, 2009

specific characterization of the mobile agents by an extension of UML 1.5 activity diagrams.

The subswinlanes modeling elements that were recently introduced in UML2.0, allowed us to define certain mobile agents characteristics: as communication and cloning. The subswinlanes are used to model the location that would be reached as a mobile agent. K.Miao and al [4] use activity diagrams to model the dynamic behavior of mobile agents through mobility, cloning and communication. They used as basis, the partitions activity notation (recently introduced in UML 2.0).

In the following section we will present our contribution which consists in the extension of UML 2.0 by mobility. This paper is an extended version of our previous conference paper [13].

3. The proposed approach

The stereotyping mechanism provided in UML2.0, allows the users (designers, analysts and others…) to add new classes of elements along with the predefined elements. A specialized element by a stereotype is semantically equivalent to a new class of metamodel which will take the same name as the stereotype.

In this section we will introduce new elements to enrich the UML 2.0 diagrams by stereotyping: that is, the use case diagram, sequences diagram, classes diagram, activity diagram, object diagrams, statechart diagram and the deployment diagram. These elements will allow us to model principal concepts like: locations, migrations and the cloning, that characterizes the mobility.

3.1 Use case diagram

3.1.1 Mobile actor:

A *mobile agent* can be represented in this diagram as a mobile actor. It is an actor stereotyped by <<ma: mobile actor>> (figure1). Like other actors of UML a mobile actor can participate in use case.

3.1.2 Migration association:

A new relation is created between any common action and other mobile through a stereotyped association by <<send>> (figure 2); it shows the original location from which the mobile agent (actor) can move.

The basis location of ***mobile actor (agent)*** is the basis platform of the ***actor*** from which the relation (arrow) starts.

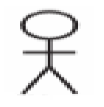


Figure 1: Mobile Actor

3-2 Sequences diagram

The cloning actions and the mobile agent movements in relation to the system objects interactions within the sequences diagrams(figure 3), as well as the locations to which this agent will move, are introduced by the following elements application:

1- <<moveTo: *id-location*>> : the mobile agent will move to the location identified by ***id-location***.

2- <<clone: nbr>> : the mobile agent creates ***nbr*** agents as the exact copies of itself.

3- <<return>> : in this case, the mobile agent must return to its original location

3-3 Class diagrams

The ***MA*** class (Mobile Agent) is an abstract class

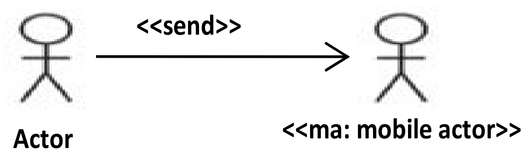


Figure 2: The migration association <<send>>

stereotyped by <<mobile>> , it should be always present while the agents mobile system is being modeled.

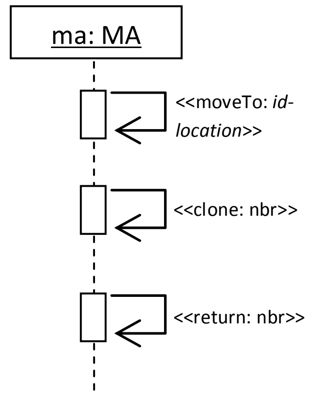


Figure 3: The new elements introduced in sequences diagram

Therefore, all mobiles classes system are inherited from these class. The class <<mobile>> gathers 3 public methods (figure4) essential to the mobile agent realization:
1-**clone(int) :** by this method, the mobile agent can create several clones of itself.
2-**moveTo(location) :** by this method, the mobile agent can move to the specified **location**.
3-**return :** by this method, the mobile agent can return to its original location.

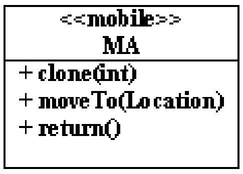


Figure 4: The mobile class

3-4 Object diagram

A mobile object is created by an instance of mobile class (figure 5). It is represented, in our object diagram, by an UML object stereotyped with <<mobile>>. we introduce the below elements in the object diagrams to show the differences between the system objects exchanges messages and the cloning and movement actions as well as the locations to which the mobile agent will move.

1- i: <<moveTo: *id-location*>> : the mobile agent will move to the location (***id-location***).
2- j: <<clone: nbr>> : the mobile agent creates ***nbr*** copies of itself.
3- k: <<return>> : in this case, the mobile agent must return to its original location.
(**i**, **j** and **k** : sequence numbers that represents the order of these actions in relation to the orders of messages exchanged (figure-6).

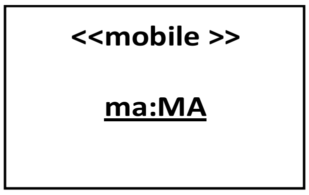


Figure 5: Mobile Object

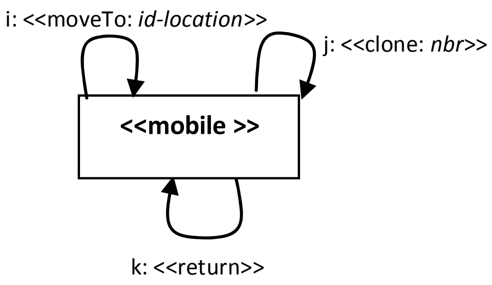


Figure 6: The new introduced elements

3-5 Statechart diagram

UML2.0 provides the multidimensional hierarchical partitions in which one dimension represents a location and another orthogonal dimension represents an object. These modeling elements are more flexible to capture new concepts of mobility. We can easily adapt them in both statechart and activity diagrams of our extension.

In the extended statechart diagram, one dimension represents a *location* that carries the stereotype <<**platform:** ***id-location***>>, which designs the basis agent platform. The Others orthogonal dimensions represent objects and mobile agents of the system to model. These mobile agents are distinguished from other objects by the use of the stereotype **<<mobile>>**(figure7).

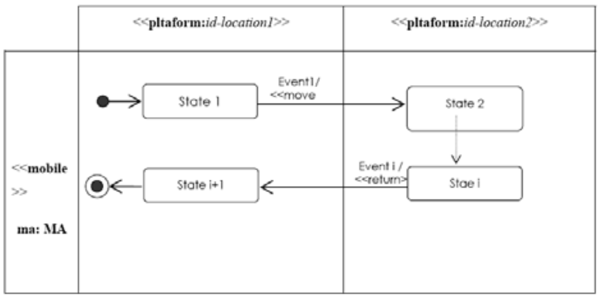
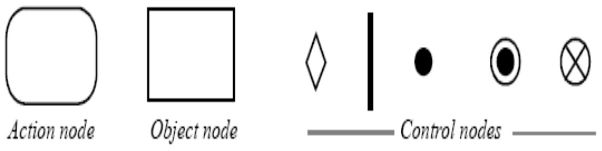


Figure 7: Statechart diagram

3-6 Activity diagram

The activity diagram holds generally three kinds of nodes: action node, object node, and control node, as shown below.



UML2.0 provides new activity nodes: the **parameter nodes** (*ActivityParameterNode*s) that are object nodes placed at the beginning and at the end of flows, to accept inputs to an activity and provide outputs from it.
We propose in our extension two modeling elements:
- A ***MobileAgentNode*** used to model the flow of a mobile agent in activity diagrams, is represented by an object node stereotyped by **<<mobile>>**.
- A ***MobileParameterNode*** used to model the flow of a mobile agent at the beginning and the end of activities, is represented by a parameter node stereotyped also by **<<mobile>>**.

In figure8: **ma2:*MA*** represents a mobile agent node, then **ma1:*MA***, **ma3:*MA*** and **ma4:*MA*** represents a mobile parameter nodes.

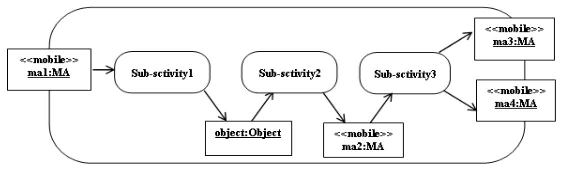


Figure 8: mobile agent and parameter nodes in activity diagram

3-7 Deployment diagram

The deployment diagram is used to model the distributed and client/server systems. This diagram shows the physical distribution of the nodes units' treatments as well as their interior components. A node in a mobile system can send and receive mobile agents in our extended deployment diagram. A node that sends mobile agents will be labeled by the stereotype **<<send>>** followed by these mobile agents names. Also, mobile nodes that receives mobile agents will be labeled by the stereotype **<<receive>>** followed by the mobile agents names.

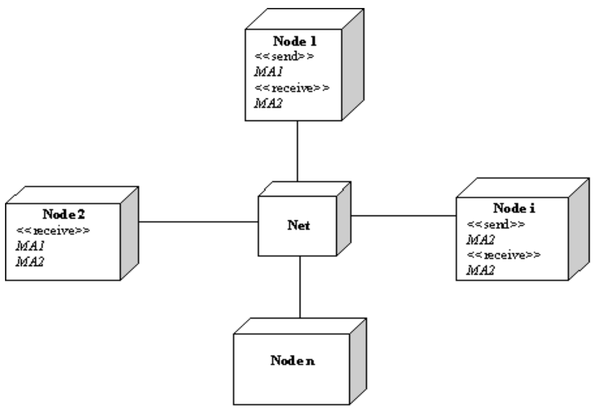


Figure 9: Deployment diagram.

In the above figure 9, the first node ***Node1*** can send the mobile agent ***MA1*** and receive the mobile agent ***MA2***, the second node ***Node2*** can receive both mobile agents ***MA1*** and ***MA2***, finally, the node ***Node i*** can send the mobile agent ***MA2*** and receive the mobile agent ***MA1***.

4. Case study: The mobile electronic Stock exchange

To realize an electronic award system with mobile agents, we suppose the existence of many shareholders (customers and sellers) each shareholder got a checking account in a bank system and a share account in stock exchange. Whenever a stock exchange session is opened, the shareholder starts to send the buying and selling orders. The system gathers two agents types (mobiles and stationary):
-The mobiles agents are organized into 3 classes: the buyers (**BMA**: buyer mobile agent), the sellers (**SMA** : seller mobile agent), and the stock exchanges (**PMA**: stock exchanges mobile agent).
-The stationary agents however, are represented by cheque account manager agents class (**CAM**: cheque account manager) each agent **CAM**$_i$ resides in the bank platform belonging to a system. Moreover, we find on the stock exchange platform two stationary agents , a share account manager agent (**SAM**: share account manager) and a transaction manager agent (**TM**: transaction manager).

During a stock exchange session , when a share buyer (**i**) sends an a buying order of particular action, an agent **BMA**$_i$ will move to a stock exchange platform containing the **TM** agent and taking orders with the following information:

- the identity of the share buyer,
- the name of the wished share,
- the claimed quantity,
- and also, the proposed price.

After the arrival of the mobile agent **BMA**$_i$ the agent **TM** classifies the buying order if it is accepted. In another way, if a share seller (**j**) sends a selling order, this provokes the movements of a mobile agent **SMA**$_j$ to the **TM** agent platform taking the selling order which got the following information:

- the identity of share seller,
- the name of the proposed share,
- the quantity and the price claimed,

- Finally, this selling order will be classified by the **TM** in the same way as that of the buying one.

In a determined time, the **TM** agent calculates the rate of each share put up for sale in the open session. Then, it classifies the best selling and buying order in according to an adopted strategy. The **TM** agent resends after that the mobile agents **BMA$_s$** and **SMA$_s$** coming from selected shareholders to confirm their respective orders. According to the case, a transaction might be confirmed or canceled.
If a transaction is affirmed, the agent **TM** will create a mobile agent **PMA** which will clone to an equal number to bank cheque account of the shareholders concerned by this transaction. Each **PMA$_i$** will move to the platform where the **CAM$_i$** agent is. As soon as it arrives, the **PMA$_i$** invokes the **CAM$_i$** corresponding to update all the counter signed accounts in order to credit the selling accounts and debit the buying accounts. Like that, the **TM** should invoke the **SAM** agent to put the shareholders share update. This **TM** agent will be responsible to calculate regularly the stock exchange index. Finally , after putting update the checking accounts on the platforms of the **CAM$_i$** agents level, each mobile agent **PMA$_i$** should retune to its basis location(the platform of the **TM** agent).

The following figure shows the global architecture of our mobile electronic stock exchange system as well as all the mobile agents.

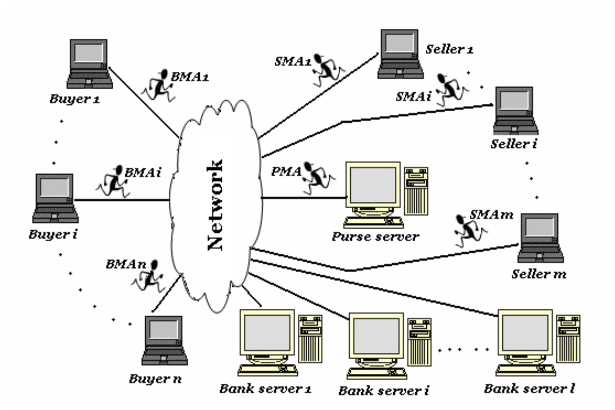


Figure 10: the mobile electronic stock exchange system

With extensions projection of UML 2.0 proposed in our approach. We present as bellow the different stereotyped UML diagrams for the electronic stock exchange mobile system.

4-1 Use case diagram

The following figure shows the use case diagram of the stock exchange system. This diagram contains in addition to the stationary actors, three mobile actors **BMA**, **SMA**, and **PMA**.

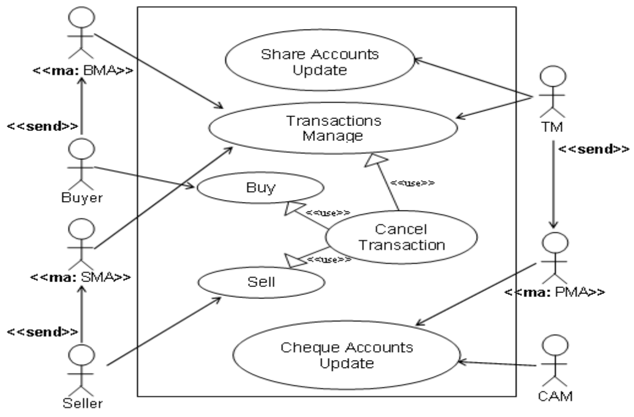


Figure 11: The use case diagram of the mobile stock exchange.

- A mobile actor **BMA**(figure 11) could be sent from a platform of a buyer to participate with the **TM** actor in the use case *Transactions Manager.*
- A mobile actor **SMA** could be sent from a seller platform to participate also in the precedent use case *Transactions Manager.*
- In the same way, a **PMA** stock exchange mobile agent could be sent from **TM** agent platform to a particular **CAM** agent platform to participate in the *Cheque Accounts update use case.*

4-2 Sequences diagram

The following sequence diagram details the use case *Buy*.

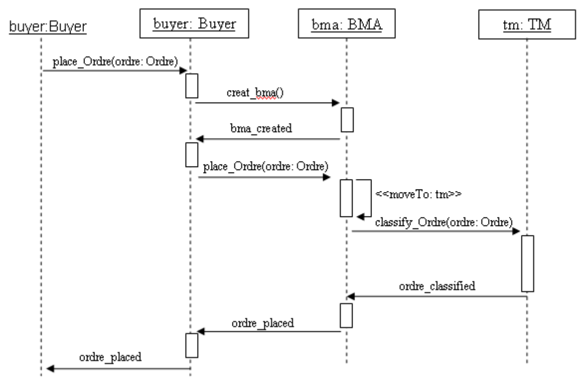


Figure 12: The sequence diagram of buying order placing.

The new element stereotyped with <<**moveTo:** ***id-location***>> allows us to capture easily the moving of any mobile agent on its time line. Consequently, we can know the locations from which the mobile agent interacts with other objects of the system. For instance, after moving to the platform of the TM agent (showed in the precedent figure by <<**moveTo:tm**>>), the mobile agent ***bma:BMA*** can interact locally with the ***tm:TM*** agent.

4-3 Class diagram

What interest us in the class diagram of the stock exchange system, is the abstract mobile class **MA** from which the three other stock exchange system mobile classes (BMA: Buyer Mobile Agent, SMA: Seller Mobile Agent and PMA: Stock exchange Mobile agent) are inherited. All these classes are stereotyped by <<mobile>> as shown in the figure13 below.

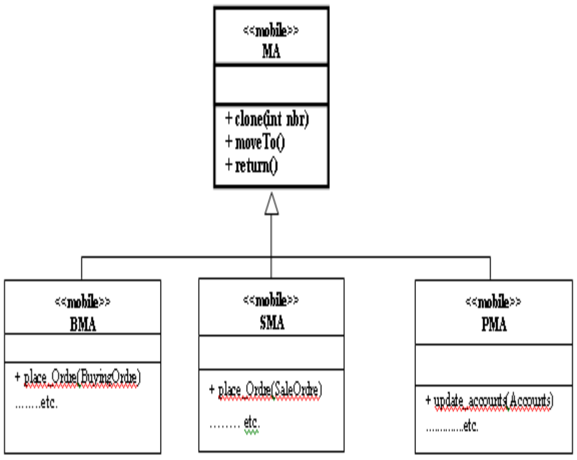


Figure 13: The stock exchange system diagram class & "the *Mobile Class"*

4-4 Object diagram

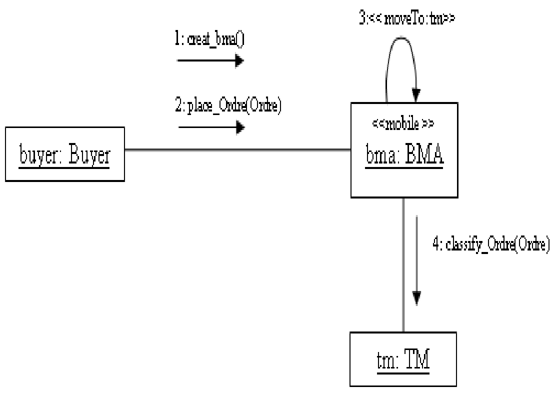


Figure 14: The object diagram of buying order placing

The above figure14 describes the sequence diagram of buying order placing (see figure 12). Then, to place a buying order :
- A ***byuer*** agent creates a byuer mobile agent ***bma*** through the message (**1: creat_bma()**);
-then, the ***buyer*** agent invoke the ***bma*** to place the buying order via the messag(**2: place_order(Order)**);
- for that, the ***bma*** should move to the basis agent ***tm*** platform, where is action is accomplished by the message (**3: <<moveTo: tm**);
- finally, the ***bma*** interact locally with the ***tm*** in order to finish its task with reference to the message (**4: classify_order(Order)**).

4-5 Statechart diagram

The following figure shows the statechart diagram of both mobile agents *BMA* and *PMA*, of our stock exchange system.

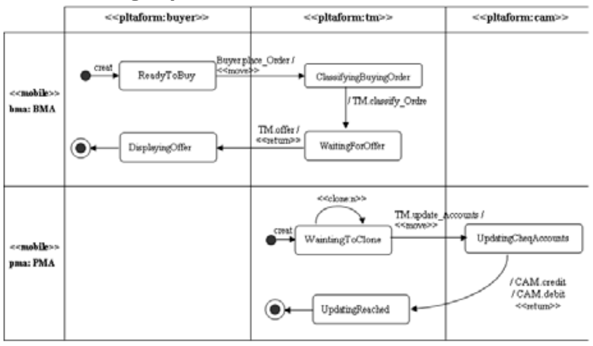


Figure 15: Statechart diagram of mobile agents *BMA* and *PMA*.

In figure15, the first orthogonal dimension shows some states of the mobile agent **<u>bma:BMA</u>** connected by transitions. The first state reachable is *ReadyToBuy;* after receiving the message *Buy.place_order* from the **<u>buy:Buy</u>** object, the ***bma*** moves to the basis platform of the *TM* agent to reach the state *ClassifyingBuyingOrder*, so it sends the message *TM.classify_Order* to the *TM* in order to classify its buying order, then the ***bma*** changes its state to *WaitingForOffer* in which it waits till it receives the message *TM.offer* from the *TM* that provokes it to move to its owner shareholder in order to display this offer. An offer consists the price calculated of the share requested and the available quantity.

3.6 Activity diagram

The figure 16 shows the activity diagram of the share buying. In the *''Place Order''* activity a buyer tries to place a buying order then a mobile agent *BMA* should move by the *''Go''* action to the basis platform of the *TM* agent. On activity level *''Receive Orders''* the *TM* receives the buying and selling orders from the mobile agents *BMA* and *SMA* arriving from their basis platforms. Two *MobileParameterNodes BMA* and *SMA* are placed in inputs of the *''Receive Orders''*. The buying order will then be managed in the *''Manage Orders''*. If this order is accepted **[Order accepted]**, the mobile agent *BMA* should return to its basis platform by the *''return''* action in order to display the current offer. In the *''Make decision''* activity, the buyer can assert **[Ok]** or cancel **[Cancel]** his order. If some buying and share orders are asserted, the *''Update Accounts''* activity will then start in order to update the cheque and share accounts of this transaction.

The rake-style linked from the interior looks like a miniature hierarchy, indicating that this activity starts another activity that represents a further decomposition.

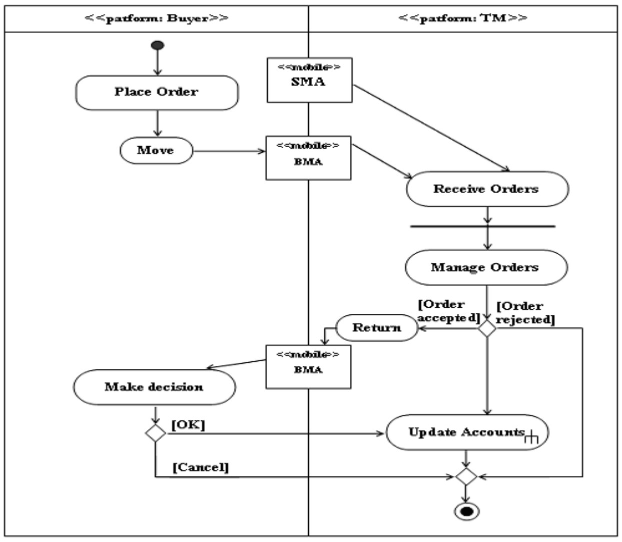


Figure 16: The activity diagram of share buying.

4-7 Deployment diagram

In this extended deployment diagram as the figure17 shows , the ***BuyerServer*** nodes could send a **BMAs** mobile agents and the ***SellerServer*** nodes could send a **SMAs** mobile agents. Whereas, the ***BancServer*** nodes could receive a **PMAs** mobile agents. Finally, the ***PurseServer*** mode could send **PMA** mobiles agents and receive **BMA** and **SMA** mobiles agents.

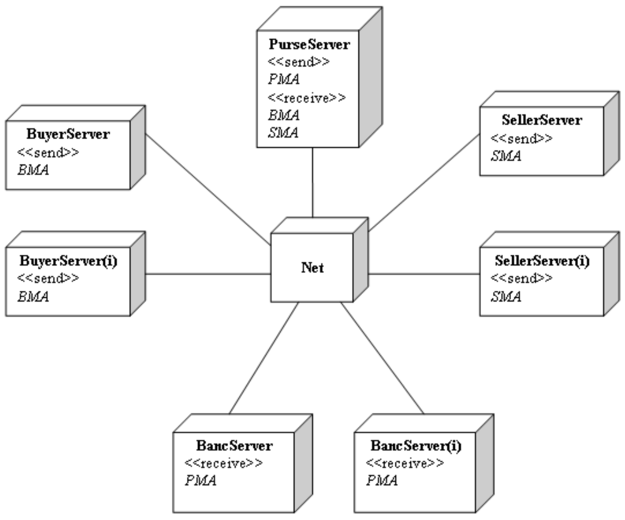


Figure 17: The deployment diagram of mobile stock exchange system.

5. Conclusion

UML2.0 offers new flexible modeling elements. They allow us to capture the mobility concepts. In this optic, we proposed in this paper, an extension of the version UML 2.0. Our objective was to introduce new modeling elements to capture the three principal concepts of mobility: the location, migration and the cloning. To show the applicability of our extension we modeled the electronic stock change as a case study.
The mobile agents' behavior is defined according to its plans and actions. The mobile agent chooses a set of plans to execute them at any moment to reach its aim. In our future work we try to introduce new modeling elements to enrich UML 2.0. So that it would be able to model the plans and the actions of the mobile agents.

References

[1] B. Bauer, J. Muller, J. Odell, Agent UML: a formalism for specifying multiagent interaction, 22nd International Conference on Software Engineering (ICSE), Agent-Oriented Software Engineering, Springer, Berlin, 2001.
[2] Foundation for Intelligent Physical Agents (www.fipa.org).
[3] M. Kang and K. Taguchi, "Modelling Mobile Agent Applications by Extended UML Activity Diagram", *Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS)'04*, Porto, Portugal, April 2004.
[4] Miao Kang, Lan Wang and Kenji Taguchi, Modelling Mobile Agent Applications in UML2.0 Activity Diagrams, 2004, http://citeseer.ist.psu.edu/668251.html
[5] D. Kinny, M. Georgeff, and A. Rao. A methodology and modelling technique for systems of BDI agents. In W. Van de Velde and J. W. Perram, editors, Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World, (LNAI Volume 1038), pages 56–71. Springer-Verlag: Berlin, Germany, 1996.
[6] MASIF: The OMG Mobile Agent System Interoperability Facility.
[7] R. Milner. Lectures on a Calculus for Communicating Systems, volume 197 of LNCS. Springer-Verlag, New York, NY, 1984.
[8] Mobile Agent Community. The Mobile Agent List. http://mole.informatik.unistuttgart.de/mal/mal.html
[9] SALEH K., EL MORR C., M-UML: An Extension to UML for the Modeling of Mobile Agent-Based Software Systems, Journal of Information and Software Technology, Vol. 46, No. 4, 2004, pp. 219-227.

[10] UML: Unified Modeling Language Specification, version2.0, OMG. available at the address: http://www.uml.org

[11] M. Wooldridge and P.Ciancarini. Agent-Oriented Software Engineering: The State of the Art In P. Ciancarini and M. Wooldridge, editors, Agent-Oriented Software Engineering. Springer-Verlag Lecture Notes in AI Volume 1957, January 2001.

[12] Dianxiang Xu and Yi Deng, Modeling Mobile Agent Systems with High Level Petri Nets, School of Computer Science Florida International University Miami, FL33199.

[13] M.R. Bahri, A. Chaoui and R. Mokhtari. Modeling of mobile agent-based systems by UML2.0, *Proceedings of the 9th international Arab Conference on Information Technology ACIT'2008*, Hammamet-Tunisia, December 2008.

Mohamed Redha Bahri is with the department of computer science, Faculty of Engineering, University Mentouri Constantine, Algeria.

Rabah Mokhtari is a PhD student in the department of computer science, Faculty of Engineering, University Mentouri Constantine, Algeria.

Dr Allaoua Chaoui is with the department of computer science, Faculty of Engineering, University Mentouri Constantine, Algeria. He received his Master degree in Computer science in 1992 (in cooperation with the University of Glasgow, Scotland) and his PhD degree in 1998 from the University of Constantine (in cooperation with the CEDRIC Laboratory of CNAM in Paris, France). He has served as associate professor in Philadelphia University in Jordan for five years and University Mentouri Constantine for many years. Dr Allaoua Chaoui has published many articles in International Journals and Conferences. He supervises many Master and PhD students. His research interests include Mobile Computing, formal specification and verification of distributed systems, and graph transformation systems.