CIT. Journal of Computing and Information Technology, Vol. 24, No. 2, June 2016, 149–163
doi: 10.20532/cit.2016.1002695

Mobile Agent-Based Software Systems Modeling Approaches: A Comparative Study

Aissam Belghiat[1,3], Elhillali Kerkouche[2], Allaoua Chaoui[3] and Mokhtar Beldjehem[4]

[1]Department of Computer Science, University of 20 August 1955-Skikda, Algeria
[2]Department of Computer Science, University of Jijel, Algeria
[3]MISC Laboratory, Department of Computer Science, University of Constantine 2, Algeria
[4]University of Ottawa, Canada

Mobile agent-based applications are special type of software systems which take the advantages of mobile agents in order to provide a new beneficial paradigm to solve multiple complex problems in several fields and areas such as network management, e-commerce, e-learning, etc. Likewise, we notice lack of real applications based on this paradigm and lack of serious evaluations of their modeling approaches. Hence, this paper provides a comparative study of modeling approaches of mobile agent-based software systems. The objective is to give the reader an overview and a thorough understanding of the work that has been done and where the gaps in the research are.

ACM CCS (2012) Classification: Computing methodologies → Artificial intelligence → Distributed artificial intelligence → Mobile agents

Keywords: mobile agent, mobile agent-based application, UML, formal method

1. Introduction

The mobile agent paradigm is an extension of client/server and remote evaluation paradigms that it provides a new development way for distributed systems. Mobility is the novel concept that characterizes the paradigm. It is the migration capability of an agent from a host to another one where it resumes its execution to achieve a task. Mobile agents present several features that provide multiple advantages of this paradigm compared to others [1]. They have a heterogeneous nature, operate asynchronously, adapt dynamically and react autonomously to changes. Furthermore, mobile agents reduce the network load and overcome its latency by acting locally. They are also robust, fault-tolerant and they encapsulate communication protocols which facilitates their upgrading.

Despite these notable advantages, the paradigm brings significant problems especially security threats which have arisen because a mobile code generated by one party will move to and execute on an environment controlled by another party. This is posing multiple issues that must be taken into account, including authentication, authorization (or access control), intrusion detection, etc. A mobile agent application can be threatened by malicious agents, malicious platforms and third parties. Malicious agents can attack agents and hosts in order to trouble them for behaving inappropriately. Analogously, malicious platforms can attack agents to do the same. Other attacks can be performed by some other entities in the network. In fact these security problems have hindered largely the development and maintenance of mobile agent paradigm, especially in critical applications such as e-commerce [2].

Modeling and designing of mobile agent-based software systems have received important attention in the last years [3]. Building a modeling language from scratch is a difficult task, this is what was inciting the researchers to borrow the essential elements of success from the object-oriented paradigm, and introduce to them the appropriate features (functionalities) in order to support the mobile agent paradigm.

This article collects, categorizes and reviews the current modeling approaches of mobile agent-based software systems which have been developed in order to deal with the analysis and design phases of those systems. We have focused on the approaches based on UML (Unified Modeling Language) [4] and considered as full languages since they cover multiple views of software systems. Furthermore, we provide a framework for the comparison of these languages that consists of multiple features identified from a bibliographical review. A thorough discussion is then carried out, that allows more comprehension of the pros and cons of multiple approaches in the literature. Consequently, readers could at the end of this article make their choices about using this or that approach or definitely proceed to define their own, based on a deep understanding.

The rest of the paper is organized as follows: in Section 2, we present some related work. In Section 3, we present some basic notions about agent technology to make the paper self-contained. In Section 4, we present our classification scheme of mobile agent-based software systems modeling approaches. In Section 5, we define a frame-work for comparing different languages. Finally, in Section 6, concluding remarks are drawn from the work and perspectives for further research are presented.

2. Related Work

Only few works are available in the literature which survey the techniques and tools for modeling mobile agent based software systems. Our paper is similar to some works, such as the one in [5] that provides a survey of existent agent-oriented methodologies. It discusses what approaches have been followed to assist in all the phases of the life cycle of an agent-based application. It is noted that the researchers have worked on extending the existing methodologies to include the relevant aspects of the agents. These extensions have been carried out mainly in two areas: object oriented (OO) methodologies and knowledge engineering (KE) methodologies. Melomey et al. [6] present in their paper an evaluation of the approaches and methodologies already proposed to develop the mobile agent paradigm. Hence, they proposed a number of required concepts to model mobility in order to overcome the limitations of the proposed approaches, and adequately model agent mobility. The authors in this paper mix between the modeling languages and methodologies. Belloni and Marcos [3] have surveyed multiple UML extensions for mobile agents modeling in order to define a unique UML extension which gathers and integrates the features of a large scale of such systems. Hachicha et al. [7] have reviewed multiple approaches which model mobile agent based applications with UML. Cervenka and Trencansky [8] have provided a survey on agent-oriented modeling techniques; however they have focused on methodologies and multi-agents systems. Melomey et al. [9] have provided a comparative study of some modeling languages for agent systems. The selected languages are actually not adequate to model mobile agents systems, but multi-agent systems. Bauer and Muller [10] have presented an overview of the existing approaches and methodologies for agent based applications. The modeling of mobile agents is not the central interest of this work.

Other approaches can be cited as well, but one issue here is worth noting, and this is the mixing of the concepts of modeling languages and methodologies in those contributions. Hence, in our paper, we have focused on modeling languages because we believe that they represent the key of success of the paradigm since mobile agents are investigated in different domain areas. We think also that methodologies cannot be general but rather adapted to specific situations. So, surveying methodologies is beyond the scope of this paper. In addition, we consider only the applications with mobile agents and static locations. Applications with dynamic locations (mobile computing) are beyond the scope of this paper.

3. Mobile Agent-Based Software Systems

3.1. Basic Concepts and Architecture

A mobile agent-based software system involves multiple agents (stationary or mobile) that interact and communicate between themselves in an adequate running environment (mobile-agent

platforms or servers) where the execution of the agent can take place and where different services and resources are provided. We present here the essential elements (factors) of this architecture [3]. Figure 1 shows the architecture of the mobile agent-based software systems.

- **Agent:** it is a software entity that executes tasks on behalf of someone (a person) or something (an organization or another agent) with some autonomy – i.e., its actions could be not only determined by external events or interactions, but also by its own motivation (or purpose).
- **Stationary agent:** it is an agent that executes in the place where it started, i.e. it does not move.
- **Mobile agent:** it is an agent that can move from one node to another to accomplish its task. It may or may not return to its "home-node".
- **Agent platform:** it is a software component where the agents can perform their tasks and where different services and resources are provided for them.
- **Node:** it is the infrastructure of execution environment on which mobile agents and mobile agent platforms are actually performed.
- **Region:** it is a logical network of places which have the same authority.
- **Place:** represents logical locations, provided by mobile-agent platforms, where agents execute, and manipulate some local resources.
- **Resources:** represent non-autonomous entities, such as files, databases, and others, which can be used and shared by several agents.

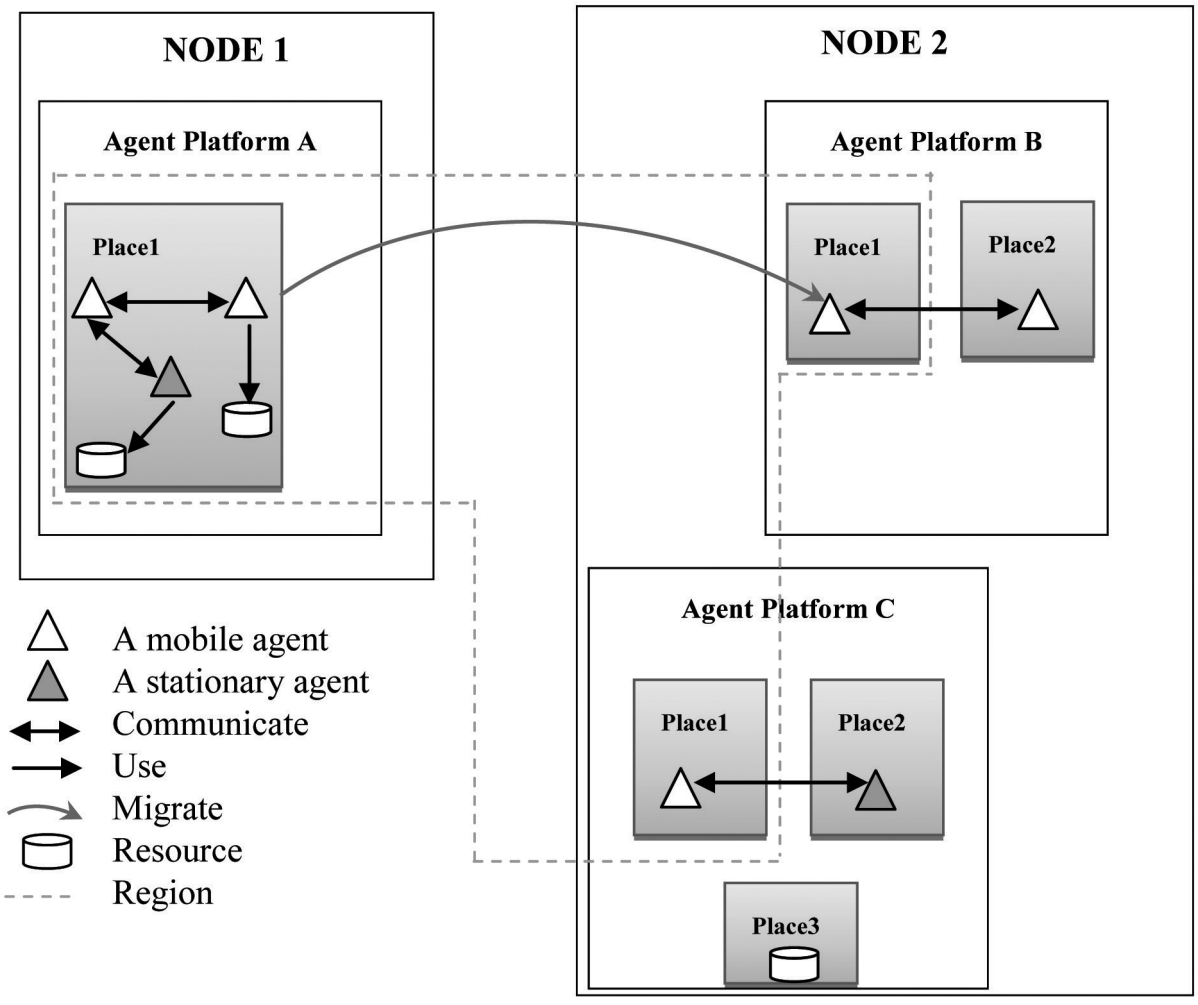

Figure 1. Mobile agent-based software systems architecture.

3.2. Mobile Agent Platforms and Applications

Proliferation of the mobile agent paradigm results in a large number of platforms. This provided multiple problems such as incompatibilities between different platforms of execution of mobile agents. It has become necessary to propose a standardization of the concepts and functionalities of the mobile agent platforms in order to ensure a high level of interoperability. Therefore, two standards have been developed by international organizations of standardization to deal with these problems; the MASIF (Mobile Agent System Interoperability Facilities Specification) standard [11] and the FIPA (Foundation for Intelligent Physical Agents) standard [12].

There are several available platforms for mobile agent development. These include, but are not limited to, Aglets [13], Ajanta [14], Grasshopper [15], Voyager [16], JADE [17], TAgent [18], Tacoma [19]. Most mobile agent platforms consider Java as the agent programming language due to its portability. The emergence of suitable mobile agent platforms has encouraged the industry to get interested in using mobile agent technology in order to develop its own products systems. These have been carried out in wide range domain areas such as [20] [1]: network management, telecommunication, electronic commerce, data mining, multimedia, climate environment, weather, e-learning, semantic web services, parallel processing, human tracking, geographic information systems and e-health monitoring systems.

3.3. Modeling Challenges and Shortcomings

Mobility is the novel concept, which characterizes mobile agent-based software systems. Its modeling represents a great challenge because it produces complicated dynamic reconfiguration of the system in question during its execution. Thus, providing an approach that has the capability of designing, this is not trivial. In fact, the approach must give answers to questions that arise from the use of mobile agents [21] such as: why does a mobile agent move from one host to another, when and where does it move and how does it reach the targeted host.

Regarding security of mobile agents, multiple researches have delivered a number of mechanisms, which follow different ways in addressing the aforementioned security issues. Some of these mechanisms have been embedded into mobile agent platforms to make them secure for network programming. Some other mechanisms can be specified by designers to enhance the security of the application. Actually, several modeling approaches provide security views which allow the description of different mechanisms used in order to prevent vulnerability of the mobile agent applications. Such mechanisms include, for instance: authentication of agents and mobile-agent systems, authorization and access control to resources. However, it is not really clear how one can define standard ideas for security modeling, because the mobile agents are applied in large number of domains which have different concerns (each domain has its own particularities).

In object oriented systems, UML [4] has played an essential and decisive role in the success of this technology. In fact, using it, you can model all parts of development of an object oriented application, from its inception to its deployment. Researchers have followed suitable approaches in the antecedent paradigm for mobile agent modeling in order to define a modeling language that has the ability to play in the mobile agent paradigm the same role that UML is already playing in the object oriented paradigm.

In general, the life cycle of development of mobile agent-based software systems is composed of three phases: the analysis phase, design phase, and implementation phase. Actually, in these systems the later phase has been enriched recently in a satisfactory way by a great number of different platforms that support and implement the paradigm of mobile agents. In contrast, the proposed approaches for the analysis and design of mobile agent-based software systems are not widespread, they are limited in scope and incomplete in themselves. Figure 2 represents the current development process lifecycle of mobile agent-based software systems.

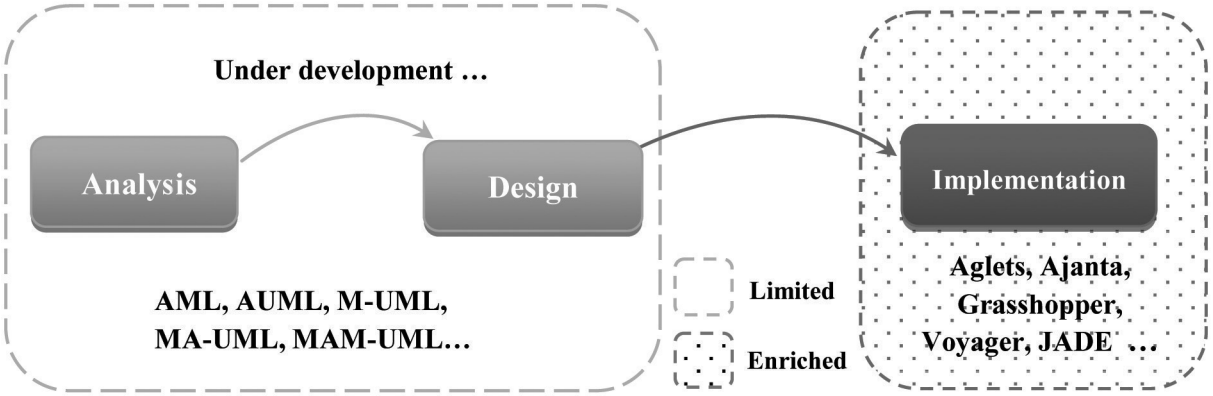

Figure 2. The life cycle of the current mobile agent-based software systems development process.

4. On Classifying Mobile Agent-Based Software Systems Modeling Approaches

Researches on providing appropriate modeling techniques for mobile agent software systems are very restricted, and they need to be improved. In this section, several modeling approaches are overviewed, and some others are just cited in order to cover large points of views on how to deal with such systems. Thus, we are able to classify existing mobile agent-based software systems modeling proposals in three large categories (see Figure 3):

- UML-based Approaches: these approaches tackle the problem of modeling mobile agent applications using UML [4]; the standard language of object oriented paradigm.
- Formal Approaches: these approaches provide formal models for mobile agent based-systems using formal methods.
- Hybrid Approaches: these approaches result from a hybrid integration of the features of the previous two categories with other contributions.

4.1. UML-Based Approaches

In this field, researchers have tried to relate mobile agent paradigm to the object oriented paradigm by extending the de-facto standard of object oriented modeling to model mobile agents. In fact, UML [4] provides some mechanisms to extend itself. It offers *stereotypes* to create new UML elements, *tagged values* to create new kinds of properties and *constraints* to add new semantics to elements. Thus, based on the expressivity and extensibility of UML, researchers have adopted it and introduced to it the appropriate artefacts in order to support the new paradigm. So, we present here some proposals

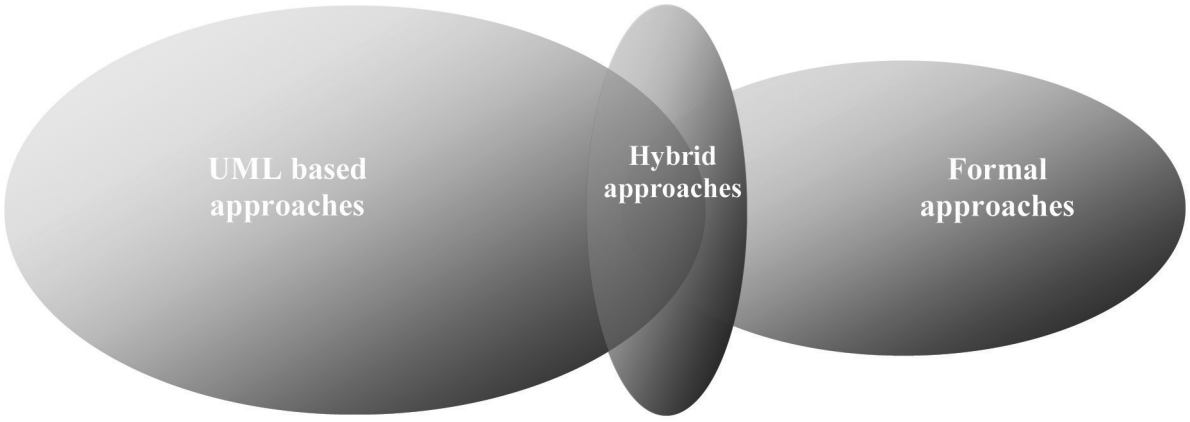

Figure 3. The three categories of existing modeling proposals for mobile agent-based systems.

that are qualified to be complete languages, i.e. which consider multiple views of the system.

- AUML (Agent UML): Bauer *et al.* [22] propose Agent UML (AUML) which can be considered as the best-known UML-based extension for agents. AUML offers the possibility of describing the internal behavior of an agent as well as its external behavior. The core parts of AUML are agent class diagrams and interaction protocol diagrams which extend respectively UML class diagrams and sequence diagrams. It was initially developed to deal only with mutli-agent systems, but its deployment and activity diagrams was extended thereafter [21], [23] to support modeling of the agent mobility. The novel notations introduced by the AUML deployment diagram extension facilitate the modeling of an agent migration to different nodes and the locations concerned by the move. Furthermore, the issue of time can be expressed by AUML activity diagrams, i.e. when a mobile agent has to migrate. Multiple concepts and notations have been developed in this language [24], but the essential features of mobile agent based applications are not taken into account. The mobility, for example, is statically described and the security is entirely omitted.
- AML (Agent Modeling Language): Cervenka et al. [8], [25] introduce Agent Modeling Language (AML); an extension of UML that incorporates different concepts and features of multi agent systems. AML is currently considered as the most complete published agent-oriented modeling language. It covers modeling of the static structure, dynamics and behavior of entities in a multi-agent system. AML attempts also to provide a modeling support for the mobility by recognizing and identifying the properties of the agent environment, host, and comportment (move and clone). The authors consider the mobility only at the implementation level when multi--agent system is deployed. This makes it unsuitable for modeling real mobile agent based applications.
- MAM-UML (Mobile-Agent Modeling with UML): Belloni and Marcos [3], [26] propose Mobile-Agent Modeling with UML (MAM-UML) to define a unique UML extension gathering and integrating the features of many UML extension-based approaches. The authors have defined a set of views and models (organizational, life cycle, interaction and mobility views) materialized by an UML profile called MAM-UML. This profile describes in details multiple interesting features of a mobile agent-based application such as itinerary of a mobile agent, the different types of movement and the tasks plan.
- M-UML (Mobile UML): Saleh and El-Morr [27] propose a complete extension of UML1.4 standard called (M-UML), where all diagrams have been extended to deal with the modeling of mobile agent-based software systems. Throughout an illustrative example, the authors introduce and describe the extensions made in each diagram UML in order to model the aspects of mobility. The authors omit the security mechanisms and their modeling. The authors focus only on the modeling of mobility feature. The approach is explained sufficiently, however its experimentation is not enough to judge its quality.
- MA-UML (Mobile Agent UML): Hachicha et al. [7] have proposed seven diagrams based on the extension of UML statechart and activity diagrams, and AUML sequence diagram. The authors claim that their contribution tackles and surmounts multiple insufficiencies not covered by the other extensions. They have developed also a software CASE Tool named MAMT (Mobile Agent Modeling Tool) [28] to support the use of MA-UML profile and the automatic generation of Java code from its diagrams which will significantly simplify the modeling and implementation of mobile agent-based applications. We have found suddenly that MA-UML is the last developed language in the recent literature (which addresses different views of mobile agent-based applications). This shows that the research community has abandoned partially this field in spite of their beneficial effects.

In addition to these languages, some proposals are very interesting. However, they represent partial approaches since they do not address all the views of a mobile agent-based software. Klein et al. [29] propose an extension to UML for mobile agents; however, their approach is limited to a specific mobile agent platform (Grasshopper platform) and to a specific class of mobile agent applications. In addition, they do not provide mobility description for all views and aspects of systems. We may also cite the works of [30] through which the authors present an approach to model the specific characterization of mobile agents by an extension of UML 1.5 activity diagrams. Bahri et al. [31] propose in their paper an extension of the most important UML 2.0 diagrams to model the mobile agent-based software systems with the objective to tackle the new concepts introduced by these systems such as migration, cloning and the locations. Kusek and Jezic [32] present a proposal for modeling agent mobility with UML sequence diagrams; they are focused on capturing agent creation, mobility paths and current agent location in order to model agent mobility.

Despite these approaches have tried to support the modeling mechanism for mobile agent-based software systems, these techniques are brought often from the field of multi-agents systems which concentrate more on the expression of the inter-agents relations. The expression of the mobility (and its results such as security) from the point of view of the distributed systems is not described enough to provide a real, efficient and robust UML-based approaches for modeling mobile agent-based software systems as in object-oriented software systems with UML.

4.2. Formal Approaches

In software engineering, formal methods are a particular kind of mathematically based techniques for the specification, development and verification of software systems [33]. Using formal approaches, to specify and to verify the functioning of mobile agent software systems during the earliest phases of the development life cycle seems to be very attractive. It must contribute to the reliability and robustness of the design, due to the power of the appropriate mathematical analysis. Actually, multiple formalisms have been used for specifying mobile agents, which are inspired strongly by two efficient formalisms; process algebras and petri nets. We recall some of them in this sub-section.

In the context of process algebras, Milner [34] proposed the π-calculus language for modeling mobile agents. It offers the possibility of link movement in virtual linked processes to describe the mobility and a wide range of the proposed approaches are based on this formalism. Jezic and Lovrek [35] have presented an approach to formal specification and verification of agent migration and communication in a mobile agent network using the π-calculus formalism. Fournet et al. [36] have proposed a calculus for mobile agents with a precise definition for migration, failure, and failure detection. They use the distributed join-calculus that is as expressive as the asynchronous π-calculus to encode their distributed calculus. This last supports explicit locations and primitives for mobility, which allows expressing mobile agents moving between nodes. Oquendo [37] proposes the π-ADL that has been designed in the ArchWare European Project (http://www.arch-ware Oquendo.org/) to address specification of dynamic and mobile architectures. It is a theoretically well-founded formal language, based on the higher-order typed π-calculus that can be used for specifying static, dynamic and mobile architectures and is illustrated through case studies. Other papers propose formalisms for description of mobile agents. Schmitt & Stefani [38] present a π-calculus extension to specify the mobility of the agents. In the paper of Sewell et al. [39], an architecture of communication between mobile agents is formalized using a π-calculus extension. Bettini et al. [33] formalize the properties of mobile agent systems, the properties of the agents are expressed dynamically according to the evolution of their localizations. Cardelli and Gordon [40] have proposed mobile ambient which is a calculus for describing the movement of processes and also devices. Other formal approaches for capturing the mobility are developed in [41] and [42].

For the purpose of specifying and verifying mobile agent systems, Petri nets and their extensions have been also widely used because these formalisms allow a clear and rigorous

representation of complex behavior of mobile agents. In this context, Xu and Deng [43] have tried to model mobile agents with higher level petri network. Xu et al. [44] have proposed in their paper an approach for formal modeling of logical agent mobility (LAM) using predicate/transition (PrT) nets. They present a mobile agent system as a set of agent spaces that is explicitly abstracted to be a component, and agents could migrate from one space to another. They present a case study of modeling and analyzing an information retrieval system with mobile agents. Also, Xu and Shatz [45] have tried to design mobile agent using G-net model. It consists of using the high level Petri net G-net model to design agents. This approach profits from the Petri net formalism which is a mature formal model in terms of graphical representation, mathematical description, and simulation tool.

4.3. Hybrid Approaches

The combination of visual modeling tools of mobile agent-based software systems and formal methods has received great attention lately. It provides powerful modeling languages for specifying and verifying these systems, taking into account the weaknesses of visual models in formal semantics due to their semi-formal character, and the weaknesses of formal models in graphical notations. Hence, in his thesis, Bahri [46] proposes a hybrid approach to model mobile agent software systems; this approach is based on an UML extension and Petri Nested Nets formalism. It takes advantage of UML, and remedies its insufficiency in terms of semantics by using Nested Petri Nets in order to enable formal analysis of these models. Knapp et al. [47] have combined a semi-formal modeling technique with formal semantics and refinement considering an extension of UML state machines for mobile objects and MTLA [48]. Latella et al. [49] have proposed an extension of UML statecharts for mobile computations accompanied with formal semantics. Belghiat et al. [50] have proposed a combined approach based on an extension of UML and the π-calculus for modeling and verifying mobile agent-based applications. Belghiat et al. [51] have also developed an automatic mapping between mobile class diagrams and OWL ontologies in order to enable automatic reasoning on these systems.

There are also other approaches which propose to use design patterns to develop mobile agent software systems such as [52], [53] and [54]. In fact, many design patterns have been identified in the literature for mobile agents such as agent patterns and interaction patterns. Some of them are described using UML while others are not.

5. Comparison Framework

5.1. Comparison Criteria

As we said at the beginning of the article, we will take interest only in modeling languages since other approaches are considered as partial contribution. In fact, to model the conceptual level of mobile agents-based software systems, multiple features and characteristics should be present in the tool developed to support analysis and design of such systems. Furthermore, Chhetri et al. [55] have proposed an interesting ontology to describe the abstract concepts and inter-relationships required to model agent mobility in order to provide conceptual level support for mobile agent applications. Analogously, Melomey et al. [6] have also proposed another ontology for the same purpose. Based on these two ontologies and by reviewing the literature essentially [3], [7], [28], and [56], we define some requirements which we believe they represent the key of success factors of each modeling approach developed for mobile agent-based applications.

- *Internal structure*; this feature illustrates the internal structure of each agent (including mobile agents) evolving in the application, i.e. its attributes such as *pro-activeness*, *reactiveness*, *autonomy*, *authority*.
- *Life cycle*; this feature determines the behavior of an agent; in other words, the *mobility path* (if it exists) and the *tasks plan* (describes the plan of tasks execution of different agents at different locations).
- *Execution needs*; this feature illustrates the execution needs for each agent (including mobile agents) evolving in the application, e.g. the *resources* and *services* they need, and the *roles* they will play.

- *Execution Environment*; this feature describes the environment of execution, i.e. *places*, *regions*, and *agent platforms*.
- *Mobility*; this is the most important feature that must be described in details by a modeling language. It is supported by the *migration*, invocation and r*emote cloning* mechanisms and it is related to the *itinerary*, and *location* concepts.
- *Interaction*; this feature illustrates the interaction mechanisms adopted in different communication acts, e.g. communication *synchronous* or *asynchronous*.
- *Security*; this feature has a major impact on the reliability of the software. The modeling language must allow describing the security mechanisms adopted, i.e. how to protect entities of the application from malicious use.

Other general features are inspired by the evaluation surveys of modeling languages, especially by the work of Berard [57].

- *Supported tools*; a good modeling language must be supported by CASE tool to facilitate their use and productivity.
- *Theoretical foundation*; a modeling language that has a solid theoretical foundation is considered as a powerful language.
- *Simplicity and clarity*; it is a feature that determines the acceptability of the language by users.
- *Process (methodology)*; a methodology that accompanies a modeling language can enhance largely the development process.

5.2. Comparison Results

Table 1 illustrates the notation we have used in our comparison, which was inspired by Crane and Dingel, 2005 [56].

Table 1. Legend of comparison.

Symbol	Description
	Unknown; not enough information to determine
	Presumably not supported
	Definitely not supported
	Partially supported
	Supported

Table 2 below summarizes the comparison of five chosen modeling languages which are defined to model mobile agent-based software systems.

Table 2. Comparison between different UML extensions for mobility.

	Modeling languages	AUML	AML	MAM-UML	M-UML	MA-UML
Basic features	Internal structure					
	Life cycle					
	Execution needs					
	Execution environment					
	Mobility					
	Interaction					
	Security					
Other features	Theoretical foundation					
	Simplicity and clarity					
	Process (methodology)					
	Supported tools					

5.3. Comparison Discussion

The table above (Table 2) summarizes the modeling capabilities of each language with regard to different features that we have proposed to check. The following lines discuss the results of the comparison:

- Internal structure of mobile agents has been taken into account entirely in MA-UML. Actually, a full diagram *"Mobile Agent Diagram"* is advocated to fully specify the internal structure of mobile agents evolved in the applications. It is an extended version of AUML class diagram which has additional properties for modeling mobile agents (e.g. authentication, history, itinerary). M-UML, in its turn, and to a lesser extent, proposes an extended version of class diagrams named as *"Mobile Class Diagram"* in order to describe the internal structure of mobile agents. AUML and AML support the internal structure of only stationary agents while MAM-UML does not support it at all.
- The lifecycle of mobile agents is supported in MAM-UML, MA-UML, and in M-UML. In fact, MAM-UML proposes a full view *"Life-cycle View"* to capture the lifecycle of the mobile agent. The view is described by using of Interaction, Statecharts and Activity diagrams. MA-UML and M-UML extend Statechart diagrams with new stereotypes and tagged values to describe all the states reached by the mobile agent during its lifetime. The feature is omitted in AUML and AML.
- Execution needs are supported completely by MAM-UML and partially supported by MA-UML. Actually, MAM-UML provides a full view *"Organizational View"* to describe the execution needs of mobile agents using Class and Object diagrams. MA-UML takes into account this feature in its *"Mobile Agent Diagram"*. The feature is omitted in AUML, AML and M-UML.
- Execution Environment is supported by AUML using an extended *"deployment diagram"*. M-UML supports also this feature using an extended version of *"component diagram"* and *"deployment diagram"*. MA-UML provides a full diagram *"Environment diagram"* to design the environment of execution, however, it represents only the static structure of the application and there is no mention about the deployment of agents. MAM-UML proposes in its *"Mobility View"* to use deployment and component diagrams to design the execution environment. The execution environment is presumably not supported by AML.
- The Mobility of mobile agents has been addressed in all the above proposed languages with varying in-depth degrees. In fact, two extensions *"activity diagram"* and *"deployment diagram"* have been defined by AUML in order to permit the modeling of mobile agent-based systems. MAM-UML provides a full view *"Mobility View"* to fully support the mobility feature. M-UML provides a full support of the mobility; it defines multiple notations and concepts to model all aspects of mobility at various views. MA-UML provides three diagrams to model the mobility; the *"Itinerary diagram"* to describe the tasks of the agent and the locations where they are performed. The *"Navigation Diagram"* which is a variant use of a *statechart* diagram to describe the mobile agent travel planning. The *"lifecycle diagram"* which is used to specify different states of the mobile agent in different locations. AML supports partially the mobility feature; it is in the deployment phase where AML proposes some notations and elements in order to take into account the mobility; however this is not sufficient because there are no details about modeling this important feature.
- Interaction feature for mobile agents is supported in MAM-UML, M-UML and MA-UML. MAM-UML provides a full view *"Interaction View"* to model this feature; the view contains collaborations and interaction diagrams. M-UML provides two extensions *"sequence diagram"* and *"collaboration diagram"* to model the mobile interactions. MA-UML provides a full diagram *"Mobile Agent Sequence Diagram"* to model interactions between different mobile agents. AUML and AML do not support mobile interactions but support simple interactions between agents.

- The security feature is partially supported by MA-UML by adding some properties (user authentication, user authorization, access rights, and authentication key) in its *"Mobile Agent Diagram"* in order to specify this feature. The other approaches do not support it at all.
- With regards to the general features which characterize a good language, there are many attempts to develop theoretical foundation, especially for AUML and M-UML. This is due to the importance given to enable the possibility of verification and reasoning about the diagrams of these languages.
- The different languages are simple to use except for MAM-UML. We cannot determine its clarity due to the lack of the experimentation (i.e. examples).
- No language between the above is combined with a process.
- AUML, AML and MA-UML are supported by CASE tools which facilitate the development of their diagrams. M-UML is presumably not supported (although, there are attempts on their statechart diagram) while MAM-UML is without any CASE tool.

In short, here is the resulting evaluation of the languages:

- We can judge that M-UML is the simplest and best adapted language for mobile agent-based software systems, which largely supports the modeling of new features (especially the mobility) of the paradigm and the discussion lines above motivate our opinion. We think that an extension of M-UML, which takes into account the above concerns, could present a complete and sufficient tool.
- MA-UML is also interesting for these systems, but the missing provision of a mechanism for modeling the architecture of the execution environment will hinder its practical use. In addition, there is a redundancy in representing information. In fact, MA-UML proposes two diagrams *"Itinerary Diagram"* and *"Navigation diagram"* to describe the same information which is the agent travel planning. Also, the combined use of the *"Navigation diagram"* and *"Lifecycle diagram"* makes the definition of another diagram *"MA Activity diagram"* redundant.
- We agree that the MAM-UML is very well organized by its different views. However, this profile has not been experimented sufficiently and it does not provide illustrative examples, which hinders largely its usage by users.
- AUML and AML are not appropriate to model mobile agent-based software systems. It is clear that the two languages were developed firstly to model multi-agent systems. After that, attempts were made to add to them some features to model mobile agent-based software systems which, we have argued, are not sufficient.

6. Conclusion

In this paper we have presented a comparative framework of the modeling approaches of mobile agent-based software systems which improves the comprehension of previous work on the topic. After a quick overview of the paradigm concepts and architecture, we present and identify challenges and shortcomings which need to be overcomed to model mobile agents-based systems and to develop powerful modeling tools for them. After that, we have proposed to classify the approaches in three large categories according to some factors: UML-based approaches, Formal approaches, and Hybrid approaches. We argue that the combination of visual (so-called informal) approaches which appeal to engineer's attention, and formal approaches, which are analyzable, are promising and likely upcoming approaches. Finally, some remarks must be drawn from our analysis and evaluation of the mobile agent paradigm and its modeling approaches:

- Mobile agents have generated considerable excitement in the research community; but they have not translated into a significant number of real-world practical applications.
- There are certainly still more approaches for modeling mobile agent-based software systems and it is impossible to recall them here.

- The proposed approaches are not complete in themselves, and their authors, to demonstrate the applicability of their approaches, present easy illustrative examples that are not complex enough to show the weaknesses of the approaches. So, it is necessary to challenge all these modeling languages against real world applications to verify their completeness.
- The combination and integration of visual modeling tools and formal methods provides powerful languages for specifying and verifying these systems, taking into account the lack of semantics in visual models and graphical notations for formal models.
- There is a pressing need to define generic meta-model and associated notations that provide a unique extensible reference that integrates all the features of the proposed approaches.
- Actually, the majority of designers have not matured sufficiently to model the mobility (and its consequences such as security) and to support the entire lifecycle of the mobile agent paradigm.
- As more researchers develop powerful tools for the modeling of mobile agent-based systems, more we can control the most challenging unsolved problems of this paradigm such as the security that is a crucial concern for such systems.
- Based on various characteristics of the mobile agents that we have presented, we can say that the mobile agents do not represent an ideal paradigm for all the types of distributed applications, but are simply a new possibility for the construction of applications that is ideal for certain domains areas and less useful for others.

Most significantly, we believe that the study above provides a clear, well defined framework for assessing various issues associated with modeling approaches of mobile agent-based software systems. This comparative framework is certainly not complete, but it has been brought to a point sufficient to serve as a basis for future refinement and extensions. Future work is to extend this evaluation to cover other categories of approaches in order to define a full reference for interested researchers in this field.

It is hoped that the paper will stimulate further work in a field whose importance will increasingly be recognized.

Acknowledgements

The authors would like to thank Mr. Rachid Echahed (CNRS and University of Grenoble, France) for his words of advice and comments regarding this work.

References

[1] D. B. Lange and M. Oshima, "Seven Good Reasons for Mobile Agents", *Communications of ACM*, vol. 42, no. 3, March 1999. http://dx.doi.org/10.1145/295685.298136

[2] L. Ma and J. J-P. Tsai, "Security modeling and analysis of mobile agent systems", vol. 57, *Imperial College Press*, 2006. http://dx.doi.org/10.1142/p435

[3] E. A. Belloni and C. A. Marcos, "Modeling of Mobile-Agent Applications with UML", in *Proceedings of the Fourth Argentine Symposium on Software Engineering*, vol. 32, pp. 1666–1141, 2003.

[4] UML (Unified Modeling Language), Object Management Group [Online]. Available: http://www.uml.org/

[5] C. A. Iglesias *et al.*, "A Survey of Agent-Oriented Methodologies", in *Proceeding of the 5th International Workshop on Intelligent Agents V: Agent Theories, Architectures, and Languages*, Springer-Verlag London, UK, 1999. http://dx.doi.org/10.1007/3-540-49057-4_21

[6] D. Melomey *et al.*, "An Evaluation of Current Approaches for Modeling Mobility of Agents", *Advances in Computing and Technology*, ICGES Press, 71–78, 2007.

[7] H. Hachicha *et al.*, "MA-UML: a conceptual approach for mobile agents modeling", *Int. J. Agent-Oriented Software Engineering*, vol. 3, no. 2/3, pp. 277–305. 2009. http://dx.doi.org/10.1504/IJAOSE.2009.023640

[8] R. Cervenka and I. Trencansky, "The Agent Modeling Language – AML: A Comprehensive Approach to Modeling Multi-Agent Systems", Springer Science & Business Media, 2007.

[9] D. Melomey *et al.*, "A Comparative Study of Modeling Languages for Agent Systems", *Systems and Information Science Notes (SISN)*, vol. 2, pp. 207–212, July 2007.

[10] B. Bauer and J. P. Muller, "Methodologies and modeling languages", in *Agent-Based Software Development*, M. Luck, R. Ashri M. D'Inverno Eds. Artech House Publishers, Boston, London, 2004.

[11] D. Milojicic *et al.*, "The OMG Mobile Agent System Interoperability Facility", *Personal and Ubiquitous Computing*, vol. 2, no. 2, pp. 117–129, June 1998. http://dx.doi.org/10.1007/bfb0057648

[12] FIPA (Foundation for Intelligent Physical Agents), "FIPA Agent Management Support for Mobility Specification", Technical report, Geneva, Switzerland, 2002, document number dc00087c.

[13] Aglets mobile agent system (2002), documentation and software [Online]. Available: http://aglets.sourceforge.net/

[14] Ajanta mobile agent system (2003), documentation and software [Online]. Available: http://www.cs.umn.edu/Ajanta/

[15] Grasshopper mobile agent system (2003), documentation and software [Online]. Available: http://www.grasshopper.de/

[16] Voyager Objectspace (2003), documentation and software [Online]. Available: http://www.recursionsw.com/products/voyager

[17] JADE (Java Agent DEvelopment Framework) (2003), documentation and software [Online]. Available: http://jade.cselt.it/

[18] TAgent (Travel Agent), documentation and software [Online]. Available: http://www.tagents.org/

[19] TACOMA (Tromsø And COrnell Moving Agents) project [Online]. Available: http://www.tacoma.cs.uit.no/

[20] A. Outtagarts, "Mobile Agent-based Applications: a Survey", *International Journal of Computer Science and Network Security*, vol. 9, no. 11, November 2009.

[21] H. Mouratidis *et al.*, "Extending the Unified Modeling Language to Model Mobile Agents", in *Proceedings Agent Oriented Methodologies Workshop, Annual ACM Conference on Object Oriented Programming, Systems, Languages (OOPSLA)*, Seattle, USA, 2002.

[22] B. Bauer *et al.*, "Agent UML: a formalism for specifying multiagent interaction", in *22nd International Conference on Software Engineering (ICSE), Agent-Oriented Software Engineering*, Springer, Berlin, 2001, pp. 91–103.

[23] A. Poggi *et al.*, "Modeling Deployment and Mobility Issues in Multiagent Systems using AUML", in *Agent Oriented Software Engineering IV*, P. Giorgini, J. P. Muller, J. Odell Eds. LNCS 2935, Springer-Verlag 2004.

[24] B. Bauer and J. Odell, "UML 2.0 and agents: how to build agent-based systems with the new UML standard", *Engineering Applications of Artificial Intelligence*, vol. 18, 141–157, 2005. http://dx.doi.org/10.1016/j.engappai.2004.11.016

[25] R. Cervenka *et al.*, "AML: Agent Modeling Language Toward Industry-Grade Agent-Based Modeling", in *Agent-Oriented Software Engineering V: 5th International Workshop*, J. Odell, P. Gioginin, and J. P. Muller, Eds. pp. 31–46, Springer-Verlag, Berlin, 2005. http://dx.doi.org/10.1007/978-3-540-30578-1_3

[26] E. Belloni and C. Marcos, "MAM-UML: An UML Profile for the Modeling of Mobile-Agent Applications", in *Proceedings of the XXIV International Conference of the Chilean Computer Science Society (SCCC'04)*, 2004. http://dx.doi.org/10.1109/QEST.2004.14

[27] K. Saleh and C. EL-Morr, "M-UML: an extension to UML for the modeling of mobile agent-based software systems", *Journal of Information and Software Technology*, vol 46, pp. 219–227, 2004. http://dx.doi.org/10.1016/j.infsof.2003.07.004

[28] H. Hachicha *et al.*, "MAMT: an environment for modeling and implementing mobile agents", in the *Sixth International Workshop From Agent Theory to Agent Implementation, at the Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08)*, Estoril, Portugal (EU), 13 May, 2008.

[29] C. Klein *et al.*, "Extension of the unified modeling language for mobile agents", in *Unified Modeling Language: Systems Analysis, Design and Development Issues*, K. Siau, T. Halpin Eds. Idea Group Publishing, pp. 116–128, 2001. http://dx.doi.org/10.4018/978-1-930708-05-1.ch008

[30] M. Kang and K. Taguchi, "Modeling Mobile Agent Applications by Extended UML Activity Diagram", in *Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS)'04*, Porto, Portugal, April 2004.

[31] M. R. Bahri *et al.*, "Towards an extension of UML2.0 to model mobile agent-based systems", *International Journal of Computer Science and Network Security*, vol. 9, No. 10, 2009.

[32] M. Kusek and G. Jezic, "Extending UML Sequence Diagrams to Model Agent Mobility", in *7th International Workshop*, LNCS 4405, Springer-Verlag, 2007, pp. 51–63. http://dx.doi.org/10.1007/978-3-540-70945-9_4

[33] L. Bettini *et al.*, "Formalizing properties of mobile agent systems", in *Coordination Models and Languages*, January 29 2002, pp. 72–87. http://dx.doi.org/10.1007/3-540-46000-4_9

[34] R. Milner, *Communicating and Mobile Systems: the π-calculus*, Cambridge University Press, 1999.

[35] G. Jezic and I. Lovrek, "Using Pi-Calculus for specification of mobile agent communication", in *IASTED Conf. on Software Engineering and Applications*, 2004, pp. 356–361.

[36] C. Fournet *et al.*, "A Calculus of Mobile Agents", in *CONCUR'96*, Vol. 1119, Springer, 1996, pp. 406–421. http://dx.doi.org/10.1007/3-540-61604-7_67

[37] F. Oquendo, "π-ADL: An Architecture Description Language based on the Higher Order Typed π-Calculus for Specifying Dynamic and Mobile Software Architectures", *ACM Software Engineering Notes*, vol. 28, no. 8, USA, May 2004.

[38] A. Schmitt and J. B. Stefani, "The kell calculus: A family of higher-order distributed process calculi," in *Global Computing*, 2004, pp. 146–178.

[39] P. Sewell *et al.*, "Location independent communication for mobile agents: A two-level architecture", in *Lecture Notes in Computer Science*, 1999. http://dx.doi.org/10.1007/3-540-47959-7_1

[40] L. Cardelli and A. Gordon, "Mobile ambients", in *Theor. Comp. Sci.*, vol. 240, 2000, pp. 177–213. http://dx.doi.org/10.1016/S0304-3975(99)00231-5

[41] R. D. Nicola *et al.*, "Klaim: a kernel language for agents interaction and mobility", *IEEE Trans. Software Eng.*, vol. 24, no. 5, 315–330, 1998. http://dx.doi.org/10.1109/32.685256

[42] T. A. Kuhn and D. V. Oheimb, "Interacting state machines for mobility", in *FME 2003*: Formal Methods, Springer Berlin Heidelberg, 2003., pp. 698–718. http://dx.doi.org/10.1007/978-3-540-45236-2_38

[43] D. Xu and Y. Deng, *Modeling Mobile Agent Systems with High Level Petri Nets*, School of Computer Science, Florida International University Miami, FL33199, 2000.

[44] D. Xu *et al.*, "A Formal Architectural Model for Logical Agent Mobility", *IEEE Transactions on Software Engineering*, vol. 29, no. 1, January 2003.

[45] H. Xu and S. M. Shatz, "A Design Model for Intelligent Mobile Agent Software Systems", Computer Science Department, The University of Illinois at Chicago, May 2002.

[46] M. R. Bahri, "Une approche intégrée Mobile-UML/Réseaux de Petri pour l'Analyse des systèmes distribués à base d'agents mobiles", *Université Constantine*, Algérie 2009.

[47] A. Knapp *et al.*, "Refining Mobile UML State Machines", in *AMAST'04*, LNCS, , Springer Verlag, 2004, pp. 274–288. http://dx.doi.org/10.1007/978-3-540-27815-3_23

[48] S. Merz *et al.*, "A spatio-temporal logic for the specification and refinement of mobile systems", in *Fundamental Approaches to Software Engineering (FASE 2003)*, vol. 2621 of LNCS, Warsaw, Poland, Springer-Verlag, April 2003, pp. 87–101.

[49] D. Latella *et al.*, "Mobile UML statecharts with localities", CNR ISTI, Pisa, Italy, Technical report 37, 2003.

[50] A. Belghiat *et al.*, "Formalization of Mobile UML Statechart Diagrams using the π-calculus: An Approach for Modeling and Analysis", in *ICIST 2014*, G. Dregvaite and R. Damasevicius Eds. CCIS 465,. Springer, 2014., pp. 236–247 http://dx.doi.org/10.1007/978-3-319-11958-8_19

[51] A. Belghiat *et al.*, "Bridging the Gap between Modeling of Mobile Agent-based Systems and Semantic Web using Meta-Modeling and Graph Grammars", in *The 7th International Conference on Information Technology (ICIT)*, Jordan, 2015. http://dx.doi.org/10.15849/icit.2015.0020

[52] Y. Aridor and D. B. Lange, "Agent design patterns: elements of agent application design", in *Proceedings of the Second International Conference on Autonomous Agents, Minneapolis*, MA: ACM Press, May 1998, pp. 108–115. http://dx.doi.org/10.1145/280765.280784

[53] R. Tolksdorf, "Coordination patterns of mobile information agents", in *Proceedings of Cooperative Information Agents II, Second International Workshop (CIA'98)*, Springer, 1998, pp. 246–261. http://dx.doi.org/10.1007/bfb0053689

[54] E. F. Lima *et al.*, "An Approach to Modelling and Applying Mobile Agent Design Patterns", *ACM Software Engineering Notes*, vol. 29, no. 4, 2004.

[55] M. B. Chhetri *et al.*, "Ontology-Based Agent Mobility Modelling", in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, Hawaii, 2006, pp. 45–54. http://dx.doi.org/10.1109/HICSS.2006.369

[56] M. Crane and J. Dingel, "On the semantics of UML state machines: Categorization and comparison", School of Comp., Queen's Univ., Technical Report 2005–501, 2005.

[57] E. V. Berard, "A comparison of object-oriented methodologies", Object Agency Inc., Technical report, 1995.

Received: May, 2015
Revised: December, 2015
Accepted: December, 2015

Contact addresses:
Aissam Belghiat
Department of Computer Science
University of 20 August 1955-Skikda
Algeria
e-mail: belghiatissam@gmail.com

Elhillali Kerkouche
Department of Computer Science
University of Jijel
Algeria
e-mail: elhillalik@yahoo.fr

Allaoua Chaoui
MISC Laboratory
Department of Computer Science
University of Constantine 2
Algeria
e-mail: a_chaoui2001@yahoo.com

Mokhtar Beldjehem
University of Ottawa
K1N 800 King Edward Avenue
Ottawa
ON K1N 6N5
Kanada 6N5
e-mail: mbeldjeh@uOttawa.ca

Aissam Belghiat is an assistant professor at the Department of Computer Science, University of 20 August 1955-Skikda, Algeria. Also, he is affiliated to the MISC Laboratory, University of Constantine 2, Algeria. His research interests include mobile agents, modeling with UML, formal methods, model transformation, systems verification, and ontoglies.

Elhillali Kerkouche is an Assistant Professor in the Department of Computer Science, University of Jijel, Algeria. His research field is UML, formal methods and distributed systems.

Allaoua Chaoui works at the Department of Computer Science, Faculty of Engineering, University Mentouri Constantine, Algeria. He received his Master's 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 an associate professor in Philadelphia University in Jordan for five years and University Mentoury Constantine for many years. During his career he has designed and taught courses in Software Engineering and Formal Methods. 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.

Mokhtar Beldjehem is a Global Citizen software engineer, educator, researcher, author, consultant, entrepreneur and philanthropist. He holds B.Eng., M.Eng., and Ph.D. in Computer Engineering, Computer Science and Software Engineering. He has been intimately involved in software engineering research since 1985 and in soft computing research since 1989. His current research interest is the interplay of soft computing and software engineering.