On the evaluation of MAS development tools

Emilia Garcia, Adriana Giret, and Vicente Botti

Abstract Recently a great number of methods and frameworks to develop multiagent systems have appeared. Nowadays there is no established framework to evaluate environments to develop multiagent systems (MAS) and choosing between one framework or another is a difficult task. The main contributions of this paper are: (1) a brief analysis of the state of the art in the evaluation of MAS engineering; (2) a complete list of criteria that helps in the evaluation of multiagent system development environments; (3) a quantitative evaluation technique; (4) an evaluation of the Ingenias methodology and its development environment using this evaluation framework.

1 INTRODUCTION

Nowadays, there is a great number of methods and frameworks to develop MAS, almost one for each agent-research group [11]. This situation makes the selection of one or another multiagent development tool, a very hard task. The main objective of this paper is to provide a mechanism to evaluate these kind of tools. This paper shows a list of criteria that allows a deep and complete analysis of multiagent development tools. Through this analysis, developers can evaluate the appropriateness of using a tool or another depending on their needs.

The rest of the paper is organized as follows: Section 1.1 briefly summarizes the state of the art of the evaluation of MAS engineering. Section 2 details some important features to develop MAS. Section 2 explains a quantitative technic to evaluate MASDKs (MultiAgent System Development Kits). In Section 3, the Ingenias methodology and it MASDK are presented. They are evaluated in Section **??**. Finally, Section 4 presents some conclusions and future works.

Emilia Garcia, Adriana Giret and Vicente Botti
Technical University of Valencia, e-mail: {mgarcia,agiret,vbotti}@dsic.upv.es

Please use the following format when citing this chapter:

Garcia, E., Giret, A. and Botti, V., 2008, in IFIP International Federation for Information Processing, Volume 276; *Artificial Intelligence and Practice II*; Max Bramer; (Boston: Springer), pp. 3544.

1.1 Background

Shehory and Sturm [9] provide a list of criteria that includes software engineering related criteria and criteria relating to agent concepts. Also they add a metric evaluation. Cernuzzi and Rossi [3] present a qualitative evaluation criteria employing quantitative methods for the evaluation of agent-oriented analysis and design modeling methods. The related works focus their efforts on the analysis of methodologies, but do not analyzes the tools that provide support for these methodologies. It is a very important feature because a well-defined methodology loses a great part of its functionality if there is no tool to apply it easily.

Eiter and Mascardi [4] analyzes environments for developing software agents. They provide a methodology and general guidelines for selecting a MASDK. Their list of criteria includes agent features, software engineering support, agent and MAS implementation, technical issues of the MASDKs and finally economical aspects.

Bitting and Carter [2] use the criteria established by Eiter and Mascardi to analyze and compare five MASDKs. In order to obtain objective results from the evaluation Bitting and Carter add a quantitative evaluation. Sudeikat and Braunch [10] presents an interesting work in which they analyzes the gap between modeling and platform implementation. Their framework allows the evaluation of the appropriateness of methodologies with respect to platforms.

This paper is based on the related works. The main objective of this paper is to offer a list of evaluation criteria that allows to analyze and compare methods, techniques and environments for developing MAS. A metric is added to the qualitative criteria to allow a quantitative comparison. These criteria focus on the gap between the theorical guidelines of the methodologies and what can be modeled in the MASDKs. Furthermore, these criteria analyze the gap between the model and the final implementation, i.e., which implementation facilities provide the MASDKs and which model elements have no direct translation in the implementation platform.

2 CRITERIA

In this section, a list of evaluation criteria is described. These features allow a complete analysis of a MASDK and the selection between one and another. They are grouped in five categories.

2.1 Concepts and properties of MAS

As it is well known, there is no complete agreement on which features are mandatory to characterize an agent and a MAS. This is the reason why an analysis of the basic notions (concepts and properties) of agents and MAS are necessary at the beginning of the evaluation. This section deals with the question whether a methodology and it associated MASDK adhere to the basic concepts and properties of agents and MAS.

- AGENT FEATURES

These features are grouped into basic features that represent the core of agenthood, and advanced features that represent specific and desirable agent characteristics.

Basic features

Agent architecture: It represents the concepts that describe the internals of an agent. The importance of this feature is not to say which approach is better than other, but this feature is very useful to know if the approach is appropriate to specific requirements.

Properties: Agents are supposed to be autonomous, reactive, proactive and socials. In this section which agent properties are supported by the methodology and by the MASDK is analyzed.

Advanced features

Mental attitudes: The agent has mental notions like beliefs, desires, intentions and commitments.

Deliberative capabilities: The agent is able to select some possible plans to solve a problem and deliberate to choose the most appropriate in this situation.

Adaptivity: The adaptability feature may require that a modeling technique be modular and that it can activate each component according to the environmental state.

Meta-management: The agent is able to reason about a model of itself and of other agents.

- MAS FEATURES

Support for MAS organizations. At this point will be analyzed only which kind of organizations are supported, the other specific characteristics of the organizations will be analyzed in the following categories.

Support for the integration with services. Some MAS software engineering has been expanded to the integration with services [8]. At this point is interesting to analyze which kind of integration is supported by the approach (agents invoke services, services invoke agents or bidirectional) and the mechanisms used to facilitate the integration. Related with this, it is very interesting to know which services communication and specification standards are supported.

2.2 Software engineering support

The development of a MAS is a complex task that can be simplified with the use of MAS engineering techniques. This section will analyze how MASDKs support this techniques.

- APPLICATION DOMAIN

There are some methodologies and MASDKs thatcan be used to develop any kind of MAS, but other approaches are specialized in a particular application domain [5].

- MODEL-CENTRAL ELEMENT

Traditionally, agents are the model-central element in most MAS models, but in the last years there are an evolution to the organization-oriented modeling and service-oriented modeling.

- METHODOLOGY

Methodologies can be analyzed using the following criteria:

Based on metamodels. Meta-model presents relationships, entities, and diagrams, which are the elements to build MAS models.

Models dependence. A high dependence on some specific models of a modelling method may imply that if they are not well-designed it may affect all the design process; hence, lower dependence is better.

Development process. It indicates which software-development process follows the methodology.

Lifecycle coverage. In complex systems such as MAS it is desirable to use tools that facilitate the development of the application throughout the entire process.

Development guides. They facilitate the developers work and make the methodology more easy to understand and follow.

Platform dependent. Some methodologies are focused on the development in a specific deployment platform.

Organization support. The methodology includes agent-organization concepts in the development life cycle.

Service support. The methodology provides support to integrate services and agents at the different stages of the life cycle.

- MODELING LANGUAGE

The methodology should use a complete and unambiguous modeling language. It can be formal, informal or a mix of them. It should be expressive enough to represent MAS structure, data workflow, control workflow, communication protocols, concurrent activities and different abstraction level views. Other advanced features are the possibility to represent restrictions in the resources, mobil agents, the interaction with extern systems and the interaction with human beings.

- SUPPORT FOR ONTOLOGIES

Ontologies represent a powerful means to organize concepts and relations among concepts in an agent-based application, and to unambiguously describe features and properties of agents. At this point if the MASDK offers the possibility to model, implement or import ontologies is analyzed.

- VERIFICATION TOOLS

The verification process can be analyzed from two points of view:

Static verification. It involves to check the integrity of the system, i.e., that the specification of all model elements and the relationships between those elements are correct. The MASDK must be able to detect inconsistencies such as an agent who pursues a goal but the functionality of the agent does not allow it to achieve that goal. Furthermore, the MASDK notifies when the modeling is incomplete (for example, when there is an objective that is not achieve for anyone). In the best cases, the application not only detects these mistakes, but also proposes solutions.

Dynamic verification. It involves testing the system using simulations, i.e., the MASDK creates a simplified system prototype and test their behavior.

- THE GAP BETWEEN METHODS AND DEVELOPMENT TOOL

This section analyzes the gap between what is theoretically defined in the method-

ology and what can be modeled by the MASDK. Three conflicted areas have been highlighted.

Complete notation The MASDK should provide the possibility to model all the methodology elements and their relationships. All the restrictions defined in the methodology should be defined in the modeling language and should be taken into account in the MASDK.

Lifecycle coverage This criterion identifies which methodology stages are supported by the MASDK.

Development guidelines These guides are very useful to develop MAS and if they are integrated in the MASDK the development task become more intuitive and easier. This integration reduces the modeling time and facilitate the development of MAS to developers.

2.3 MAS implementation

This section analyzes how the MASDK helps the developer to transform the modeled system into a real application.

- IMPLEMENTATION FACILITIES

Graphical interfaces It represents the possibility to generate graphical interfaces using the MASDK.

Limited systems The MASDK may support the development of system with some limitations, i.e., the development of system that are going to be executed in limited devices like mobile phones.

Real time control Some application need real time control, so it must be supplied for the MASDK.

Security issues The MASDK can provide security mechanism to ensure that agents are not malicious and do not damage other agents, that their agents are not be damaged and has to avoid the interception or corruption of messages. These issues are more complex when the system has mobile agents or when agents interact with extern agents.

Physical environment models These are a library of simulators of physical parts of some kinds of systems for testing.

Code implementation The MASDK allows to implement or complete the agents code in the same tool.

Debugging facilities They are necessary for developing correct, reliable and robust system, due to the complex, distributed and concurrent nature of MAS.

- THE GAP BETWEEN MODELING AND IMPLEMENTATION

Match MAS abstractions with implementation elements

Agents use abstract concepts that are close to those used when reasoning, about human behaviours and organizations. This fact can facilitate the analysis and design activities but the gap between model and implementation increases.

Automatic code generation

It is an important feature because it reduces the implementation time and the errors in the code.

Code language This issue represents which programming language is used to generate agent code and which language is used to represent the ontologies.

Platform This issue represents for which agent platform is generated the code.

Generation technology Nowadays, there are a great number of techniques and languages to transform automatically from models to code.

Kind of generation It can be complete or it can generate only the skeletons of the agents. These skeletons usually have to be manually completed for the developer.

Utility agents There are different agents offering services that do not depend on the particular application domain (for example yellow and white pages). The MASDK should also provide them.

Reengineering These techniques are very useful in traditional software engineering and also in MAS development.

Services If the generating MAS is integrated with services, the MASDK should provide the necessary mechanisms for this integration.

2.4 Technical issues of the MASDK

This criteria selection is related to the technical characteristics of the development environment. Some of these features can have a dramatic impact on the usability and efficiency of this tools.

Programming language The language used to implement the MASDK and the language used to store the models are important keys.

Resources System requirements to the MASDK which include in which platforms can be executed and if it is light-weight.

Required expertise It indicates if it is necessary be a expert modeler and developer to use the MASDK.

Fast learning It indicates if the MASDK is easy to use and does not need much training time.

Possibility to interact with other applications For example this can provide the possibility to import or export models developed with other applications.

Extensible The MASDK is prepared to include other functional modules in an easy way.

Scalability This issue analyzes if the MASDK is ready to develop any scale of applications (small systems or large-scale applications).

Online help A desirable feature in a MASDK is that it helps developers when they are modeling or implementing, i.e., the MASDK takes part automatically or offer online suggestions to the developer.

Collaborative development This functionality may be very interesting to develop complex systems in which there are a group of developers which cooperates.

Documentation An important aspect when dealing with new proposals is how they are documented. A good documentation and technical support should be provided.

Examples If the MASDK presents complete case study is another feature to evaluate. The fact that the MASDK has been used in business environments also demonstrate the usefulness of the MASDK.

2.5 Economical aspects

Economical characteristics are important to choose between one or another MASDK. Obviously, one key in the evaluation is the cost of the application, the cost of it documentation and a technical service is provided. Also, the vendor organization gives an idea about the reliability and the continuity of the application.

2.6 Metric

A numerical evaluation offers a fast and general evaluation which allows to compare and evaluate methods and tools easily.

Each established criterion in Section 2 is associated with a weigh that represents the importance of this criterion. A ranking of 0 indicates that this criterion cannot be quantitative evaluated. For example, the use of one agent architecture or another cannot be evaluated as better or worst, it is only a feature and it will be more or less appropriate depending on the requirements of the system to develop. A ranking of 1 indicates that this criterion is desirable but not necessary. A ranking of 2 indicates that it is not necessary but very useful. A ranking of 3 indicates that it is necessary or very important in the MAS development. An evaluation vector for each MASDK is created stating how the approach covers each criterion. The scale of the points is 0, 25, 50, 75 or 100% depending on how feature is covered. The numerical evaluation is the result of the dot product between the weight vector and the evaluation vector.

The presented metric is based on [2] although this metric does the average of all the criteria without separating categories. In this paper the numerical evaluation is considered taking into account the categories established in Section 2 to detect which parts of the MASDKs has more lacks and should be improved.

	Concepts and properties of agents and MAS										
	Agent basic features					Agent advanced features				MAS features	
	Agent architecture	Propierties				Mental attitudes	Deliberative	Meta-management	Adaptability	Organizations	Services
		Autonomy	Reactivity	Proactivity	Sociability						
Weight	0	3	3	3	3	3	2	1	1	2	1
Qualitative ev.	BDI	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	No
Numeric ev.	----	●	●	●	●	●	◕	○	○	◕	○

Note: ○ is 0% ◔ is 0-25% ◒ is 25-50% ◕ is 50-75% ● is 75-100%

Fig. 1 Concepts and properties of MAS evaluation.

	Softare engineering support																		
	Domain	Central element	Methodology							Modeling language			Ontologies	Verification		Gap Methods-Model tool		
			Meta models	Models´ dependence	Development process		Platform dependent	Organization support	Service support	Type	Expressiveness			Static	Dynamic	Notation	Lifecycle coverage	Guidelines
					Lifecycle coverage	Guidelines					Basic	Advanced						
Weight	---	---	3	2	3	2	2	2	1	---	3	2	3	3	2	3	3	2
Qualitative ev.	General	Organizations	Yes	High	Analysis, Design, some of Implementation	Yes	Independent	Yes	No	Informal	Yes	No	No	Yes	No	No gap	Yes	No
Numeric ev.	---	---	●	◒	◕	◕	●	◕	○	---	●	○	○	◕	○	●	●	○

Fig. 2 Software engineering support evaluation.

3 CASE STUDY: INGENIAS

INGENIAS [7] is a methodology for the development of MAS that is supported by an integrated set of tools, the INGENIAS Development Kit (IDK) [6]. These tools include an editor to create and modify MAS models, and a set of modules for code generation and verification of properties from these models. In this section Ingenias and the IDK are presented and evaluated according to the framework and the metric presented in Section 2.

- CONCEPTS AND PROPERTIES OF MAS

Ingenias agents follow a BDI architecture and have all the basic properties defined by Wooldridge. Also Ingenias specifies mental attitudes and deliberative capabilities but it does not provide support to specify adaptivity, meta-management or emotionality (see Figure 1). Ingenias specifies an organizational model in which groups, members, workflows and organizational goals are described. Ingenias does not model social norms or the dynamic of the organization, i.e., how an agent can enter in a group, etc. At this moment, Ingenias does not support service integration.

- SOFTWARE ENGINEERING SUPPORT

Ingenias is a general application domain and its model-central element are organizations (Figure 2). It integrates results from research in the area of agent technology with a well-established software development process, which in this case is the Rational Unified Process (RUP). This methodology defines five meta-models that describe the elements that form a MAS from several viewpoints, and that allow to define a specification language for MAS. These metamodels are: Organization meta-model, Environment meta-model, Tasks/Goals meta-model, Agent meta-model, Interaction meta-model. These metamodels have strong dependences and relations. Ingenias provides a development guides for the analysis and design stages. These guides include organization support and have no platform dependences. Furthermore, Ingenias provide some support for the implementation stage. Ingenias also provides an informal modeling language that provides mechanism to define all the basic features explained in Section 2 but that it is not useful to represent mobil agents and the other advanced-related features. The IDK offers a module that realize a static verification of the modeled system. It is very useful because it helps developers to find errors in their models and suggests possible solutions. As is shown in Figure 5, Ingenias fills well the gap between its methodology and the development tool. The elements of the methodology and the notation of the modeling language is totally supported by the IDK. Also all the stage that are covered by the methodology, are covered by the IDK as well. At this moment the IDK has no development guidelines integrated, but there are some researches about this topic.

- MAS IMPLEMENTATION

Figure 5 shows that Ingenias does not provide a good support to the implementation stage. It has a module to transform models into Jade agents [1]. This module generate skeletons of the agents and their tasks. Almost all the elements of the methodology can be directly transformed into elements of the Jade platform, but there are some abstraction like goals or mental states that have not a corresponding element of the platform.

	MAS implementation															
	Implementation facilities							Gap modeling-implementation								
									Automatic code generation							
	Interfaces	Limited systems	Real Time	Security	Physical environments	Code impleme ntation	Debugging	Matching	Code language	Platform	Technology	Kind	Utility	Reengineery	Services	
Weight	1	1	1	1	1	1	1	3	---	---	---	2	1	1	1	
Qualitative ev.	No	No	No	No	No	No	No	Yes	Java	Jade	Templates	Skeletons	No	No	No	
Numeric ev.	○	○	○	○	○	○	○	◔	---	---	---	◒	○	○	○	

Fig. 3 MAS implementation issues of the MASDK evaluation.

	Technical issues of the MASDK										Economical aspects			
	Programming language	Requirements	Required expertise	Fast learning	Interactions	Extensible	Online help	Collaborative	Documentation	Examples	Vendor	Cost	Updates	Technical service
Weight	0	1	2	2	1	2	2	1	3	3	---	2	3	3
Qualitative ev.	Java / Xml	Multiplatform	Yes / not very intuitive	Medium	no	Yes	No	No	Yes	Yes	Grasia	Free	Yes	Yes
Numeric ev.	-----	●	◒	◒	○	●	○	○	●	◔	---	●	●	◔

Fig. 4 Technical issues of the MASDK evaluation.

Metric evaluation		Score (percent)		Comentaries
Concepts and properties of agents and MAS	Agent basic features(3)	100,00	82,86	Obtains a good result because the Ingenias methodology supports all the basic concepts and features of agents and also offers support to organizations.
	Agent advanced features(2)	90,00		
	MAS features(2)	50,00		
Software engineering support	Methodology(3)	71,67	53,93	Ingenias provides a well-defined methodology which is well-supported by its modeling language. Furthermore, it obtain a good result in the category "Gap methods-tools", so this methods are well-supported by its IDK and almost there is no gap between what is defined by the methodology and what is represented with the MASDK.
	Modeling language(3)	60,00		
	Ontology(2)	0,00		
	Verification(3)	45,00		
	Gap methods-tool(3)	75,00		
MAS implementation	Implementation facilities(3)	22,50	17,50	The MAS implementation is an ongoing work topic for the Ingenias developers. At this moment, it is not well implemented by the IDK and there are many issues that there are not covered.
	Gap model-implementation(2)	10,00		
Technical issues of the MASDK		60,29		The IDK is technically well-defined, but there are some interesting features that are not covered.
Economical aspects		90,63		The result is very good, the reason is that Ingenias is free and is well-supported by its authors.

Fig. 5 Numerical evaluation results.

- TECHNICAL ISSUES OF THE MASDK

The IDK has been implemented in Java, because of that it is multiplatfom. A new functionality module can be added to the IDK easily. The IDK is not very intuitive, but it does not require much time to learn it. The documentation of Ingenias and of its IDK is complete and some validated examples are provided with the IDK. Despite this, the examples are not completely developed and some of them have modeling mistakes.

- ECONOMICAL ASPECTS

Ingenias is an academical work developed in the Grasia! research group. This project is still open, so they are offering new versions of the IDK and improving the methodology. Ingenias and the IDK are open source and all their documentation is publicly available. There is no specific technical service but authors answer questions by email and by the gap repository.

- NUMERICAL EVALUATION

Figure 5 shows the numerical evaluation results. The final result of each category is the dot product between the weight of the features analyzed (the number inside the parentheses of the second column) and their numerical result (third column). Developers can obtain a fast overview about Ingenias and its IDK looking this figure.

4 CONCLUSION

This paper summarizes the state of the art in the evaluation of methods and tools to develop MAS. These studies are the base of the presented evaluation framework. This framework helps to evaluate MASDKs by the definition of a list of criteria that allows to analyze the main features of this kind of systems. This list covers traditional software engineering needs and specific characteristics for developing MAS. This study helps in the evaluation of the gap between the methods and the modeling tool, and the gap between the model and the implementation.

A quantitative evaluation method is presented. It allows a numeric evaluation and comparison. It gives developers a fast overview of the quality of the evaluated MASDK in each category. The weight of the criteria is a variable feature that affect the result of the evaluation. This feature provides developers a mechanism to adapt the evaluation framework to their own requirements.

This evaluating framework has been used successfully to evaluate Ingenias and its IDK. The results of the evaluation shows that this approach covers the entire development process in a basic way, but, it has important lacks in the transformation from models to the final implementation. It should improve it implementation coverage. As future work, this framework will be used to evaluate and compare a large set of MASDKs and their methodologies.

Acknowledgements This work is partially supported by the TIN2006-14630-C03-01, PAID-06-07/3191 projects and CONSOLIDER-INGENIO 2010 under grant CSD2007-00022.

References

1. F. L. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*. John Wiley & Sons, 2007.
2. E. Bitting, J. Carter, and A. A. Ghorbani. Multiagent System Development Kits: An Evaluation. In *Proc. of the CNSR*, pages 80–92, 2003.
3. L. Cernuzzi and G. Rossi. On the evaluation of agent oriented modeling methods. In *In Proceedings of Agent Oriented Methodology Workshop*, 2002.
4. T. Eiter and V. Mascardi. Comparing environments for developing software agents. *AI Commun.*, 15(4):169–197, 2002.
5. A. Giret, V. Botti, and S. Valero. *MAS Methodology for HMS*, volume LNAI 3593, pages 39–49. Springer Verlag., 2005.
6. J. Gomez-Sanz and J. Pavon. Ingenias development kit (idk) manual, version 2.5. 2005.
7. J. Pavon, J. Gomez-Sanz, and R. Fuentes. *The INGENIAS Methodology and Tools*, volume chapter IX, page 236276. Henderson-Sellers, 2005.
8. M. P. Singh and M. N. Huhns. *Service-Oriented Computing Semantics, Processes, Agents*. John Wisley and Sons Ltd, 2005.
9. A. Sturm and O. Shehory. A framework for evaluating agent-oriented methodologies. In *AOIS*, volume 3030 of *LNCS*, pages 94–109. Springer, 2003.
10. J. Sudeikat, L. Braubach, A. Pokahr, and W. Lamersdorf. Evaluation of agent-oriented software methodologies examination of the gap between modeling and platform. *AOSE-2004 at AAMAS04*, 2004.
11. M. Wooldridge and P. Ciancarini. Agent-Oriented Software Engineering: The State of the Art. In *AOSE01*, volume 1957/2001 of *LNCS*, pages 55–82. Springer, 2001.