MAS-CommonKADS: a comprehensive agent-oriented methodology*

Carlos A. Iglesias[1**], Mercedes Garijo[2],
José C. González[2] and Juan R. Velasco[2]

[1] Dep. de Teoría de la Señal, Comunicaciones e Ing. Telemática, Univ. de Valladolid, E–47011 Valladolid, Spain
{cif@tel.uva.es}
[2] Dep. Ing. Sist. Telemáticos, Univ. Politécnica de Madrid, E–28040 Madrid, Spain
{mga,jcg,juanra}@gsi.dit.upm.es

ABSTRACT

This article presents the models of an agent-oriented methodology called *MAS-CommonKADS*. This methodology extends the knowledge engineering methodology *CommonKADS* with techniques from object-oriented and protocol engineering methodologies. The methodology consists of the development of seven models: *Agent model* that describes the agents; *Coordination model* that describes the interactions between software agents; *Communication model* that describes the interactions between human and software agents; *Expertise model* that describes the reasoning process of the agents; *Task model* that describes the tasks that the agents carry out; *Organisation model* that describes the social structures between human and software agents; and *Design Model* that describes the design decisions about the agent architecture and agent network. The main advantages of the application of this methodology are: (1) a clear software development life cycle for multiagent development; (2) graphical and textual notation for each model; (3) project management through the definition of landmark states in each model; (4) usage of standardised languages for describing the conversations between the agents; and (5) usage of knowledge engineering techniques for knowledge acquisition and knowledge reutilisation.

KEYWORDS

multiagent systems, software agents, methodology, knowledge engineering, coordination.

INTRODUCTION

Agent technology has reached a level of maturity for it to be applied in industry. Nevertheless, the application of agent technology outside of the laboratories needs the definition of agent-oriented methodologies that guide the development process and the management of agent-based applications.

This article presents an agent-oriented methodology called *MAS-CommonKADS*, that extends the models of the knowledge engineering methodology *CommonKADS* (Schreiber et al., 1994) for dealing with multiagent issues.

THE MODELS OF *MAS-COMMONKADS*

MAS-CommonKADS defines the following model set (Fig. 1):

- *Agent model (AM):* specifies the agent characteristics: reasoning capabilities, skills (sensors/effectors), services, agent groups and hierarchies (modelled in the organisation model).

* This research is funded in part by the Commission of the European Community under the ESPRIT Basic Research Project *MIX: Modular Integration of Connectionist and Symbolic Processing in Knowledge Based Systems*, ESPRIT-9119, and by the Spanish Government under the CICYT projects TIC91-0107 and TIC94-0139

** This research was partly carried out while the first author was visiting the Dep. Ingeniería de Sistemas Telemáticos (Universidad Politécnica de Madrid)

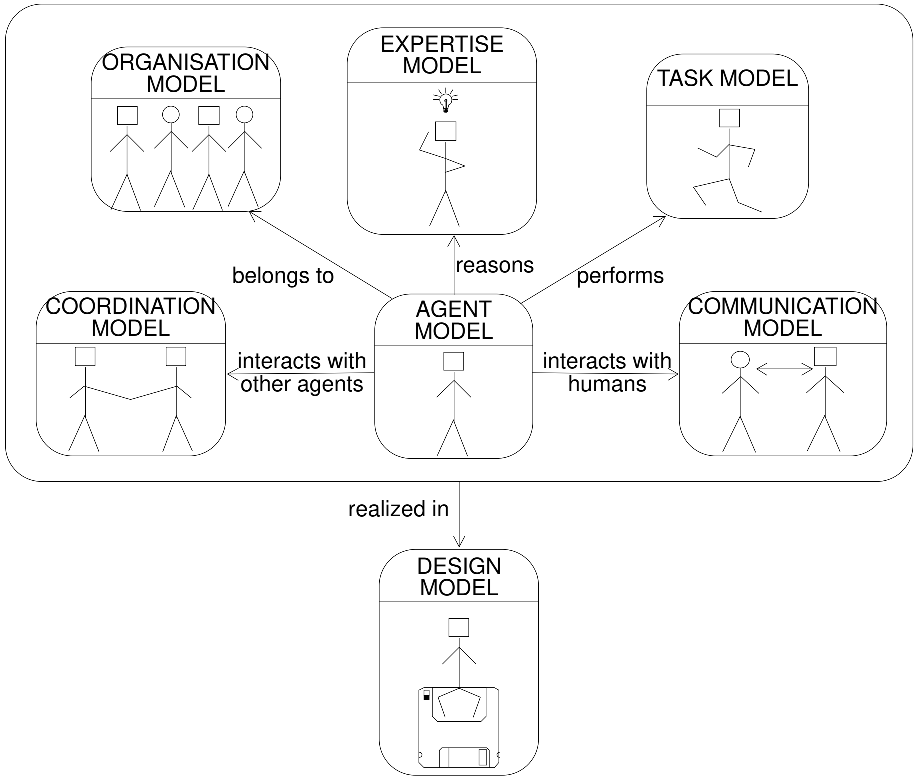

Fig. 1. Models of *MAS-CommonKADS*

- *Task model (TM):* describes the tasks that the agents can carry out: goals, decompositions, ingredients and problem-solving methods, etc.
- *Expertise model (EM):* describes the knowledge needed by the agents to achieve their goals.
- *Coordination model (CoM):* describes the conversations among agents: their interactions, protocols and required capabilities.
- *Organisation model (OM):* describes the organisation in which the MAS is going to be introduced and the organisation of the agent society.
- *Communication model (CM):* details the human-software agent interactions, and the human factors for developing these user interfaces.
- *Design model (DM):* collects the previous models and subdivides it into three submodels: *application design:* composition or decomposition of the agents of the analysis, according to pragmatic criteria and selection of the most suitable agent architecture for each agent; *architecture design:* designing of the relevant aspects of the agent network: required network, knowledge and telematic facilities and *platform design:* selection of the agent development platform for each agent architecture.

The application of the methodology consists of the development of the different models.

DEVELOPING A CASE STUDY

We will show the main phases of the methodology developing a case study: an intrusion detection system. The purpose of the system is to protect the systems against potentially harmful intruders. The system will collect data, and based on the analysis of the data, will determine what actions should be taken when a user of the system is suspected of being an intruder.

Conceptualisation

The conceptualisation phase is performed to obtain a first (and informal) description of the problem. We identify the main users of the systems and the role(s) carried out by each user of them. This is shown using the notation of use cases of OOSE (Object Oriented Software Engineering)(Jacobson et al., 1992) (Fig. 2). The interactions of the use cases are formalised with MSC (Message Sequence Charts)(Rudolph et al., 1996) as Regnell(Regnell et al., 1996) (Fig. 3).

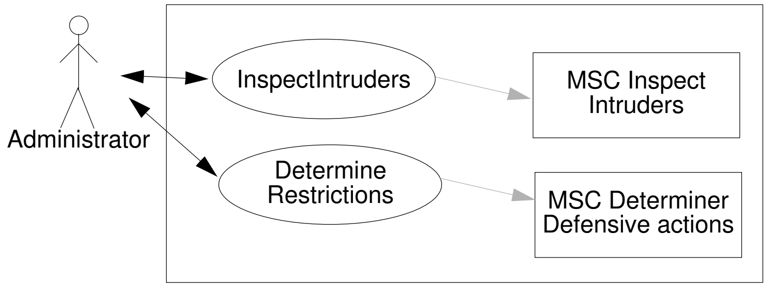

Fig. 2. Initial use cases

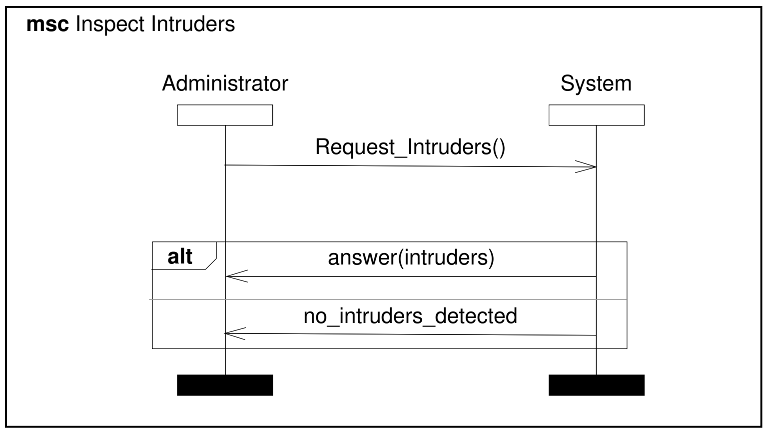

Fig. 3. Interactions of a use case

Analysis

The analysis phase develops all the models except the design model. The models can be developed concurrently, following a risk-driven development process. The objectives of this phase are the identification and description of:

– the agents by developing the agent model;
– the goals of the agents by developing the task model;
– the conversations between the agents by developing the coordination model;
– the interactions between human and software agents by developing the communication model;
– the knowledge needed by the agents to achieve their goals and the available problem solving methods by developing the expertise model; and
– the structural relationships between human and software agents by developing the organisation model.

The main steps of the analysis are: delimitation of the multiagent system from the external systems, decomposition of the system into agents, development of the models for describing the agents and cross-validation between models.

Initial task model In this case study, an initial task model can show the main tasks of the system (see Fig. 4). The task model consists of a set of coherent activities that are performed to achieve a goal in a given domain. The required knowledge to complete a task is included in an expertise model.

Agent identification The system can be decomposed attending to the identification of a conceptual distance(Bond and Gasser, 1988): geographical distance, logical distance and knowledge distance. In our case, we could consider a geographical distance for collecting the events across the network. From the initial task model, we can recognise different knowledge domains: classification of events and planning and scheduling of the actions to prevent the harmful actions of an intrusion.

Now we can refine the set of identified agents using a technique called *recursive use case modelling* (Fig. 5). For each identified use case, we establish which agent will interact with this use case, and which other agents this agent needs to "use" to carry out the task of the use case.

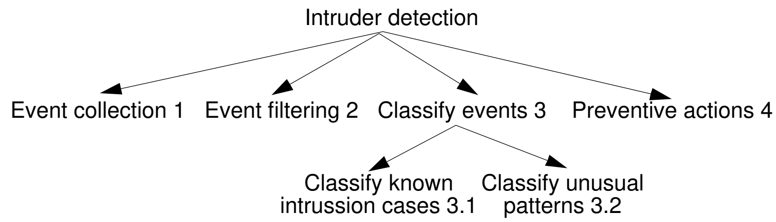

Fig. 4. Initial task model

Based on the use cases and the task model, we can identify several agents: an *Interface Agent*, that communicates the multiagent system with the user; a *Collector Agent*, whose goal is to collect the events generated by the users and filter them; a *Filter Agent*, whose goal is to classify whether the events are suspicious and a *Police Agent*, whose goal is to establish which preventive actions should be carried out to prevent harmful actions from the suspicious users. These agents could be further decomposed in sub-agents, reassigning their goals.

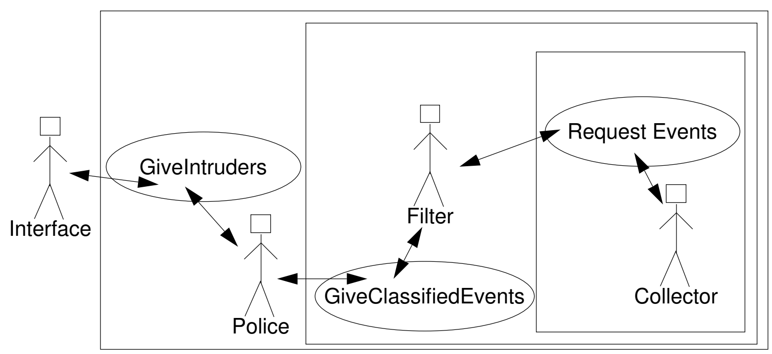

Fig. 5. Recursive use case modelling

Development of the rest of the models The expertise model describes the domain knowledge of the agents, their inferences, the order of these inferences and the problem solving methods to carry out a task. We should describe the main concepts, their properties and relationships (i.e. intruders, intrusions, network systems, etc.). For describing the inferences, the *CommonKADS* library gives the initial inference structure and related models. For example, a first approach to filter the events can be to use heuristic classification, whose inference structure diagram is shown in Fig. 6.

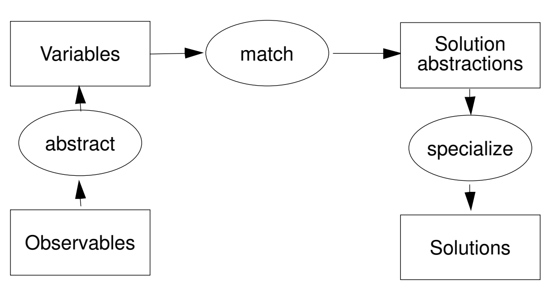

Fig. 6. Inference structure for Heuristic Classification(Tansley and Hayball, 1993)

For space limitations, the rest of the models are not developed here:

– The organisation model is used to model the human relationships in the administration group and the software agent relationships (Fig. 7). The relationships with objects of the environment are also modelled here, using extended OMT (Object Modelling Technique) notation(Rumbaugh et al., 1991).

– The communication model describes the interactions between human and artificial agents, here we should design whether the interaction will be based on menus, etc. and the content of these interactions.
– The coordination model describes the conversations between the agents. Each interaction is described using speech-acts and formalised with MSCs. The processing of each interaction is expressed in communicating extended finite state machines of the formal description technique SDL (Specification and Description Language)(ITU-T, 1994). The first version of the model supposes no existing conflict, for testing the set of identified conversations. Then, each conversation is reviewed and more complex protocols can be identified (e.g. negotiation, contract net, etc.).

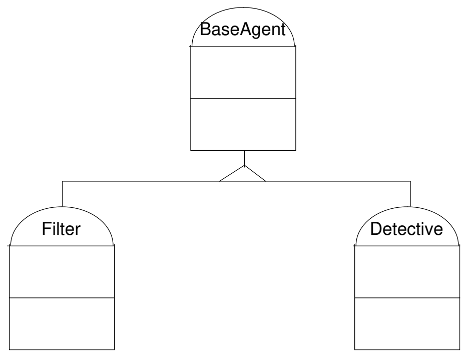

Fig. 7. Example of organisation of software agents

Design

The design consists of collecting all the functional requirements described in the previous models and the non-functional requirements. We identify several sub-tasks for developing the design model:

– *Platform design:* selection of hardware and software. We will use the MAST multiagent development tool(Iglesias et al., 1996)[1].
– *Application design:* details the requirements of each model. The results are expressed in ADL (Agent Definition Language)(Gonzlez et al., 1994), a language for describing agent services and skills that can be directly compiled.
– *Architecture design:* details the *network agents* for maintaining the agent network(Iglesias et al., 1996). Different agent networks are distinguished: for coordination (group managers, allowed interactions and protocols), for knowledge management (ontology servers, knowledge representation translators) and for network infrastructure (agent name servers, security services, yellow pages servers, etc.). In this application, the agent infrastructure is managed by a specialised agent called *YP* that acts as agent name server, yellow pages server and group manager. The usage of ontology servers will be studied during the testing phase.

Implementation and testing

The intrusion detection system is being implemented under Solaris 2.5 in a university network (Gmez Cid, 1997). The methodology does not cover the implementation phase, since it is very multiagent platform dependent, but our experience is that the implementation can be derived easily from the developed models.

The testing of the platform can take advantage of the use cases defined in the model. The automatic generation of test cases from these use cases and diagrams from the coordination model is subject of further research.

[1] The *MAST* platform is available at *http://www.gsi.dit.upm.es/~mast*

RELATED WORK AND CONCLUSIONS

Our approach offers a novel combination of knowledge engineering techniques, object-oriented techniques and formal description techniques. The advantages are threefold: easy learning of the proposed graphical notations and development steps, knowledge task libraries and easy interaction modelling. Other approaches that are based only on object-oriented techniques, as Kinny(Kinny et al., 1996), Burmeister(Burmeister, 1996), Kendall(Kendall et al., 1996) and *MASB*(Moulin and Brassard, 1996), lack of suitable techniques for knowledge modelling. In the same way, the methodology *CoMoMAS*(Glaser, 1996), based only in *CommonKADS*, lacks of graphical notation for developing the different models.

The research presented here is ongoing. Current work on *CommonKADS* includes the development of an integrated knowledge engineering tool for the application of the methodology. Other aspects of current work are to extend the library of generic tasks of *CommonKADS* for intelligent agents and to extend MSC for modelling multicast interactions.

References

Bond, A. H. and Gasser, L. (1988). An analysis of problems and research in DAI. In Bond, A. H. and Gasser, L., editors, *Readings in Distributed Artificial Intelligence*, pages 3–36. Morgan Kaufmann Publishers: San Mateo, CA.

Burmeister, B. (1996). Models and methodology for agent-oriented analysis and design. In Fischer, K., editor, *Working Notes of the KI'96 Workshop on Agent-Oriented Programming and Distributed Systems*. DFKI Document D-96-06.

Glaser, N. (1996). *Contribution to Knowledge Modelling in a Multi-Agent Framework (the CoMoMAS Approach)*. PhD thesis, L'Universtité Henri Poincaré, Nancy I, France.

Gmez Cid, C. (1997). Sindi: Sistema multiagente para deteccin de intrusos. Master's thesis, E.T.S.I. de Telecomunicacin. Universidad de Valladolid.

Gonzlez, J. C., Velasco Prez, J. R., Iglesias, C. n., Alvarez Toledo, J., and Escobero, A. (1994). A multiagent architecture for symbolic-connectionist integration. Technical Report MIX/WP1/UPM/3.0, Dep. Ingeniera de Sistemas Telemticos, E.T.S.I. de Telecomunicacin, Universidad Politcnica de Madrid.

Iglesias, C. A., González, J. C., and Velasco, J. R. (1996). MIX: A general purpose multiagent architecture. In Wooldridge, M., Müller, J. P., and Tambe, M., editors, *Intelligent Agents II (LNAI 1037)*, pages 251–266. Springer-Verlag: Heidelberg, Germany.

ITU-T (1994). Z100 (1993). CCITT specification and description language (SDL). Technical report, ITU-T.

Jacobson, I., Christerson, M., Jonsson, P., and Övergaard, G. (1992). *Object-Oriented Software Engineering. A Use Case Driven Approach.* ACM Press.

Kendall, E. A., Malkoun, M. T., and Jiang, C. (1996). A methodology for developing agent based systems for enterprise integration. In Luckose, D. and C., Z., editors, *Proceedings of the First Australian Workshop on DAI*, Lecture Notes on Artificial Intelligence. Springer-Verlag: Heidelberg, Germany.

Kinny, D., Georgeff, M., and Rao, A. (1996). A methodology and modelling technique for systems of BDI agents. In van der Velde, W. and Perram, J., editors, *Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World MAAMAW'96, (LNAI Volume 1038)*. Springer-Verlag: Heidelberg, Germany.

Moulin, B. and Brassard, M. (1996). A scenario-based design method and an environment for the development of multiagent systems. In Lukose, D. and Zhang, C., editors, *First Australian Workshop on Distributed Artificial Intelligentce, (LNAI volumen 1087)*, pages 216–231. Springer-Verlag: Heidelberg, Germany.

Regnell, B., Andersson, M., and Bergstrand, J. (1996). A hierarchical use case model with graphical representation. In *Proceedings of ECBS'96, IEEE International Symposium and Workshop on Engineering of Computer-Based Systems.*

Rudolph, E., Grabowski, J., and Graubmann, P. (1996). Tutorial on message sequence charts (MSC). In *Proceedings of FORTE/PSTV'96 Conference.*

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. (1991). *Object-Oriented Modeling and Design.* Prentice Hall.

Schreiber, A. T., Wielinga, B. J., Akkermans, J. M., and Van de Velde, W. (1994). CommonKADS: A comprehensive methodology for KBS development. Deliverable DM1.2a KADS-II/M1/RR/UvA/70/1.1, University of Amsterdam, Netherlands Energy Research Foundation ECN and Free University of Brussels.

Tansley, D. and Hayball, C. (1993). *Knowledge-Based Systems Analysis and Design.* BCS Practitioner Series. Prentice-Hall.