UDC: 004
Original scientific paper

A SURVEY OF THE PROPERTIES OF AGENTS

Kornelije Rabuzin, Mirko Maleković, Miroslav Bača
University of Zagreb, Faculty of Organization and Informatics, Varaždin
{kornelije.rabuzin, mirko.malekovic, miroslav.baca}@foi.hr

Abstract: *In the past decade agent systems were considered to be as one of the major fields of study in Artificial Intelligence (AI) field. Many different definitions of agents were presented and several different approaches describing agency can be distinguished. While some authors have tried to define "what" an agent really is, others have tried to identify agents by means of properties which they should possess. Most authors agree on these properties (at least basic set of properties) which are intrinsic to agents. Since agent's definitions are not consistent, we are going to give an overview and list the properties intrinsic to an agent. Many different adjectives were attached to the term agent as well and many different kinds of agents and different architectures emerged too. The aim of this paper it go give an overview of what was going on in the field while taking into consideration main streams and projects. We will also present some guidelines important when modelling agent systems and say something about security issues. Also, some existing problems which restrict the wider usage of agents will be mentioned too.*

Keywords: *agent, multi-agent system.*

1. AGENT DEFINITIONS

Many different definitions of the term agent can be found in the literature. One could easily be confused when looking at these definitions and different approaches when defining the term agent. That's why we have listed some of those definitions in order to show that things aren't black and white. To provoke the reader we will refer to [44] where authors wrote that a human can be seen as an agent who can operate in complex environment by combining reactive and deliberative reasoning. If humans are treated as agents, we find it reasonable to disclose and clarify this inconsistency when describing this term.

Agent is a system with the following characteristics [50]:

- Agent lives in artificial world *S* together with other agents,
- Agent can perceive and act in his environment,
- Agent possesses a representation of the world *S*,
- Agent is goal-oriented and can plan his actions and

- Agent can communicate with other agents.

According to [1] agent is a software component able to perceive its environment and react to it.

Agents are autonomous, persistent (software) components that perceive, reason, communicate and act in someone's favour, influencing its environment [12].

In [22] the term agent has been used in the sense of "processing entity" in cooperative process-oriented environments (CPE).

An intelligent agent is a computational entity with a mental state of its beliefs and goals [26].

An agent is a software entity with a well-known identity, state and behaviour, with autonomy to somehow represent its user [46]. An *agent-based application* (or ABA for short) is a dynamic, potentially large-scale distributed application in an open and heterogeneous context such as the Internet [46].

If we try to define an agent through its mental state, then certain mental components should be part of an agent: beliefs, perceptions, memory, commitments, expectations, goals and intentions [51]; this is some sort of an extension to well known and described BDI architecture (which will be explained later on), but in authors opinion perceptions and memory should also be present because perceptions (although they are not as stable as beliefs), form the basic for reactive behaviour and are hence more fundamental then for example desires or intentions [51].

Intelligent agents continuously perform three functions: perception of dynamic conditions in the environment; action to affect conditions in the environment; and reasoning to interpret perceptions, solve problems, draw inferences, and determine actions [24, 51].

In [28] authors gave an interesting look and they have defined an agent as "Agent = Logic + Architecture" (based upon Kowalski's Algorithm = Logic + Control).

Agents are logic programs that continuously perform the *observe-think-act* cycle [15].

An agent is a computer system that is situated in some environment and that is capable of autonomous action in this environment in order to meet its design objectives; intelligent agent is autonomous, reactive, proactive and capable to communicate with other agents [38].

The behaviour of an intelligent agent can be presented as a cycle [32] (*n* represents time required for one cycle):

*To **cycle** at time T,*
***observe** any inputs at time T,*
***think**,*
***select** one or more actions to perform,*
act,
***cycle** at time T+n.*

As one can see, many different definitions are presented; according to [16] it is still not clear what an agent really is; Shoham says that each usage of the term agent should also have a reference to the author and the meaning of this word.

If one looks more carefully, one can see that three main approaches when defining the term agent can be distinguished; in the first approach an accent is put on **what** an agent really is, the second approach is characterised by means of the **properties** he should possess and the third approach is the mixture of two already mentioned approaches.

Although some people think agents are similar (or equal) to the concept of an object, other disagree. The next chapter will try to remove the vague and clear up the confusion.

2. AGENT VS. OBJECT

Although many authors treat agents and objects as the same thing, there are some crucial distinctions between them which will be pointed out.

First of all, although encapsulation is immanent to both agents and objects, agents can not make orders to each other (in order to do something); they exchange messages and not orders, and they decide whether they will do something autonomously or not [18]. Similar claims can be found in [45].

According to [56] an agent is similar to an object to that extant that he possesses some state and methods which are capable of changing this state. According to the same source, the main distinctions between an agent and other software products are communication, task delegation, personalization, mobility and capabilities of doing something.

According to [51] agents and objects are not the same thing; agent possess believes and commitments while objects don't have state's generic structures. Also, the language that agents used in order to communicate is independent on any application, while objects exchange messages implemented in the language itself.

As can be found in [19], an agent is a software entity which exhibits autonomy, social awareness, reactivity and proactivity. Also, an agent represents a specialization of a traditional object; some mental components are introduced as believes, intentions and possibility to decide. While traditional objects communicate using messages, agents can differentiate several different types of communication (offer, accept, inform, request, reject, assist and so on).

Now when we have listed different definitions regarding agents and clarified the distinction between an agent and an object, let us look some properties intrinsic to agents.

3. AGENT PROPERTIES

While some authors have described the set of properties an agent should possess, others have tried to define properties which MultiAgent System (MAS) or Agent Based Application (ABA) should posses. We can say that MAS is a set of (independent) agents which operate together in order to solve some problems.

Autonomy as a property means that agent works alone and can make decisions on his own, while proactivity refers to the exploration of new possibilities and taking the initiative. Adaptability means that agent can adapt to the changing environment. Mobility refers to the capability to move through the network.

Reactivity is very important characteristic because it ensures the real time response to the proactive (dynamic) environment. Proactive environment can change independently on whether the agents are performing some actions or not [21].

Agent-based applications have certain characteristics: autonomous (each user creates and maintains his agents), heterogeneous (different programming languages, databases, communication packages and so on are used), open (agents may depend on other agents), dynamic (agents are added and removed dynamically), robust (errors will have to be tolerated) and secure (different levels of security should be implemented) [46].

Sometimes it's impossible that one agent solves a problem because of some constraints which may exist. So communication, cooperation and coordination are unavoidable [18]. Althoug in MAS agents cooperate and work together, conflicts sometimes can not be avoided (for example scarce resources could cause a conflict). According to [5] cooperation consists of collaboration and coordination (Figure 1). Conflicts among agents have been well defined and described in [50].

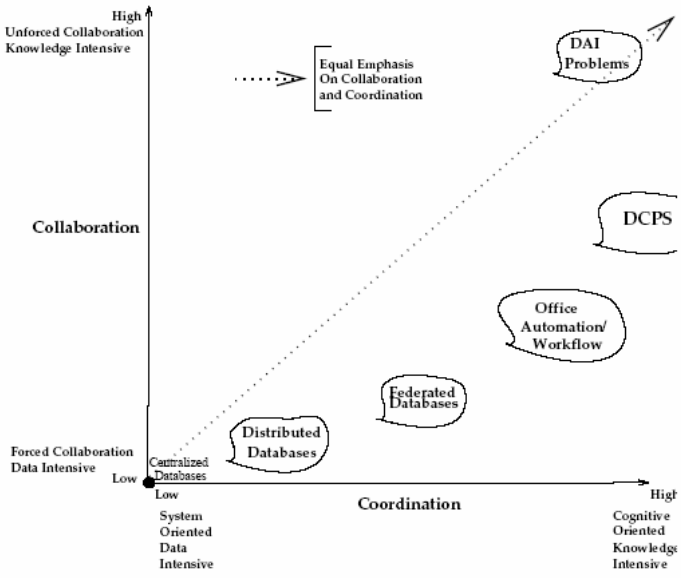

Figure 1. Cooperation [5]

There are several different ways of communication [18]:

1. *Broadcast system* – messages are sent to all agents,
2. *Blackboard system* – there is a place where agents write bids and offers,

3. *Direct communication* – an agent has to know addresses of other agents and
4. *Federated system* – there is a mediator in charge of communication.

One of the main features is definitely the possibility to communicate; that's why several different communication languages were presented. Most famous languages are KQML and ACL; nowadays it is XML too. KQML is both a message format and a message-handling protocol to support run-time knowledge sharing among agents [54]; KQML can be used as a language for an application program to interact with an intelligent system or for two or more intelligent systems to share knowledge in support of cooperative problem solving. Task sharing means that a problem has to be decomposed into several sub-problems which have to be solved and results integrated again. KQML has a basic set of performatives which determine the legal operations between agents. Performatives have certain parameters and some basic performatives are:

Table 1. KQML – basic performatives *tell* and *deny* [54]

tell	**Deny**
:content <expression>	:content <performative>
:language <word>	:language KQML
:ontology <word>	:ontology <word>
:in-reply-to <expression>	:in-reply-to <expression>
:force <word>	:sender <word>
:sender <word>	:receiver <word>
:receiver <word>	

ACL is also based on a set of performatives (for example *inform*). More on ACL can be found in [18, 38]. In the next chapter some types of agents will be identified.

4. TYPES OF AGENTS

Many different types of agents were introduced during the years; some of them were widely accepted (for example deliberative or reactive), while others were not.

The difference between an agent and other software products are: personalization (a user can determine agent's strategy), delegation (agent has to do something for the user), communication (with other agents), mobility, ROI (agent should earn more then he spends) and agent possesses some skills [55].

Regarding mobility, authors agree that agents can be divided into mobile and static agents. Mobile agents can move some piece of code and desired data through the network [47]. In [35] architecture was built by means of which agents move across the network and communicate with other agents. In order to achieve mobility, an agent server is needed. This is in fact a program which is in charge of the agents running on that computer; it can accept other agents and provide the resources for their execution. HTTP was chosen as the infrastructure because it is well understood and POST and GET methods can be used for sending agents and

status requests and so on. Mobile agents travel from one place to another, although in an open system communication problems could arise.

Deliberative agents act in accordance with BDI architecture. According to [18] deliberative agent possesses 3 main characteristics:

1. He possesses a knowledge base which contains knowledge about the environment,
2. Agent functions based on Observe-Decide-Act principle and
3. Decisions are based upon deduction or mathematical (logical) problem solving.

A reactive system is a state-based system (state change is performed) which, due to internal or external events, continuously interacts with its environment [7]. Reactive agent reacts to some events which may occur, and by reacting agent doesn't examine what happened in the past [38]. They act according to stimulus-response model.

In [20] two types of agents are differentiated: cognitive and reactive. According to [20] cognitive agent possesses some knowledge base which contains all necessary data and procedures which affect the way the agent is going to perform certain actions regarding its environment; they are mostly goal oriented and perform a set of actions in order to fulfil the given goals.

A *vivid* agent is a software-controlled system whose state is represented by knowledge base, and whose behaviour is represented by means of action and reaction rules [52]; the basic functionality of a vivid agent comprises a knowledge system (including an update and an inference operation), and the capability to represent and perform actions in order to be able to generate and execute plans.

Many different kinds of agents were presented too: in [4] agents were classified as *user agents, guide agents, autonomous agents and anthropomorphic agents*. *Desktop* agents were presented in [8]; they are executed locally and they notice the user about the results of their reactions. Functionally, we can distinguish between *interface* agents, *learning* agents, *emotional* agents and some other kinds of agents. *Adaptive* agents learn by observing and perceiving their environment [34]. These kinds of agents will not be explained; we refer to [4, 8, 34].

5. AGENT ARCHITECTURES

One has to have in mind that since the mid 1980s different architectures for reasoning agents have emerged. Some of these models were never widely accepted or realized in a form of application that could perform something and these effects could be measured. Different architectures were (are) used for the design and development of individual agents. Each architecture describes how the agent's modules operate and how can an agent interact with the environment [**Error! Reference source not found.**].

Four different types of architectures were presented in [38]:

1. Logic based – deduction is used to make decisions,
2. Reactive agents – situation to action mapping,

3. BDI architecture and
4. Layered approach – each layer is responsible for some actions.

BDI architecture is an exception and was (is) used very much [12]; it describes and determines the agent behaviour according to information an agent possesses, goals which needs to be performed and reasoning about these goals and available information:

Goals are expressed as conditions over some interval of time and are described by applying various temporal operators to state descriptions. Plans describe how an agent should react when certain facts are added to its belief database, or when it newly acquires certain goals [3]. Some relationships between knowledge operator, belief operator, and desire operator in MASs are described in [37].

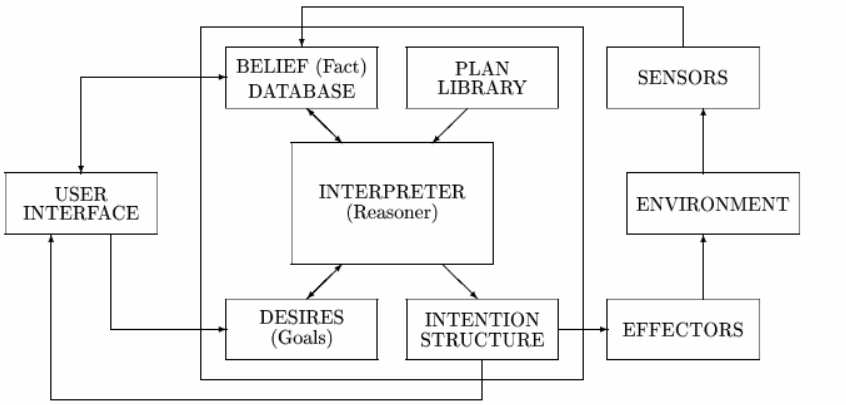

Figure 2. BDI architecture [3]

Deliberative agents have to think and plan; this is a time consuming process although it can be simple. This is not acceptable if an agent is placed in very dynamic environment which changes very fast and the reaction has to be performed quickly; so reactive architectures are considered for such cases. Reactive architecture never became really popular for agents modelling, although it was good in robotics for example; namely, it's intrinsic to agents that they are reactive. This is the reason that pro-active architectures were more popular. These architectures have certain properties in common: logical correctness, pro-activity, ability to learn, adapt, react and interact [**Error! Reference source not found.**].

It's hard to determine to what extent should an agent exhibit deliberative and to what reactive behaviour; it depends on number of parameters like tasks, architecture, and application domain and so on. In [38] it is pointed out that it is very hard to build a well balanced agent system.

An interesting attempt was made in [32] and [33] where authors have tried to reconcile the rational and reactive agent architectures; the behaviour of a rational agent was delineated by means of an abstract procedure which represents agent's observation-thought-action cycle. Similar procedure was defined for reactive agent too. It was obvious that these two cycles differ in some respects and unified agent cycle was presented; every iteration of an observe-think-act cycle can be used for

making observations and learning new facts and rules from the environment. The behaviour of an intelligent agent was characterised in simplified terms as a cycle [32] (as it was already shown in the first chapter):

*To **cycle** at time T,*
***observe** any inputs at time T,*
***think**,*
***select** one or more actions to perform,*
act,
***cycle** at time T+n.*

Some authors also distinguish the notions of strong and weak agency. According to [56], weak agency subsumes the following properties: autonomy, social ability, reactivity and pro-activity. There is less agreement on the strong agency term, but it subsumes knowledge, belief, intention and obligation.

According to [6] intelligent behaviour consists of reactive and deliberative behaviour (Figure 4):

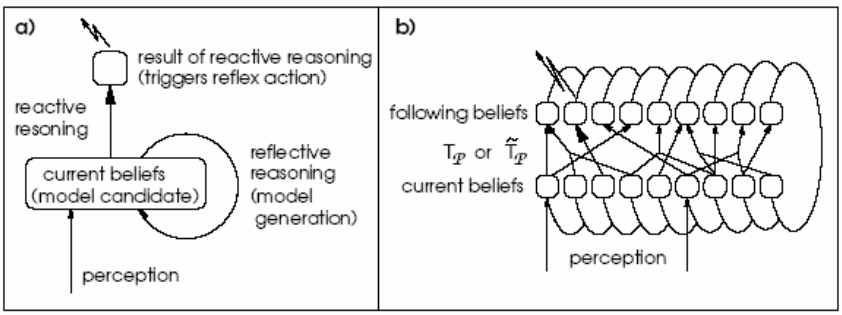

Figure 3. Reactive reasoning depending on current beliefs (a) and the same architecture where the results of reactive reasoning influence agent's beliefs (b) [6]

In [28] the idea of hybrid agents has been presented; a hybrid agent is the one that integrates several different styles like reactive, deliberative and cooperative. According to [8**Error! Reference source not found.**] agent needs to possess 4 modules (parts): machinery, content, security and access. A three-tiered architecture has been proposed in [44]; it combines reactive and deliberative behaviour. Logic-based deliberative module endows the agent with the ability to "*think ahead*" and behaviour-based reactive module ensures that the agent can handle various real-time challenges of the environment [44].

6. AGENTS MODELLING

The process of agents modelling is not an exception; several different approaches have been presented and many tools are used for implementation. One of the approaches can be seen on the following figure:

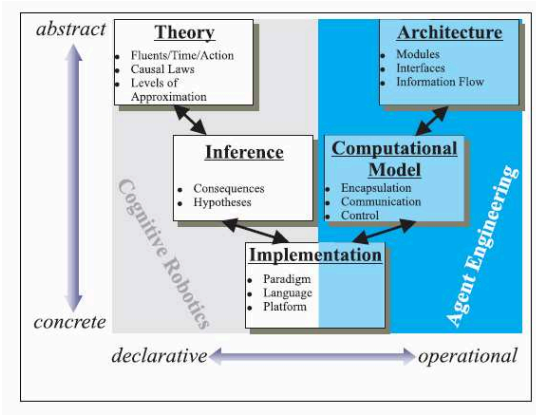

Figure 4. Agents modelling [28]

In order to implement agents and multiagent systems, many programming languages are used as well as special shells developed for that purpose. One of the widespread languages used for implementation is Java [18, 20, 52, 55]. Also, Prolog, C/C++ and Smalltalk are used very much too. Some of the shells developed for agent implementation are ARIS [36], 3APL [26], SIM_AGENT [14], Simula++ [11] and so on.

For modelling agents' actions in [21] TEAL (Temporal Executable Action Language) is used. Action rules have the form:

$$\textit{past literal} \wedge \textit{exec(a)} \rightarrow \textit{future literals}$$

In [30] a part of formal methods, X-machines, are used for agent modelling. This method can model agents through their behaviour. X-machine is a type of FSA (Finite State Automata) and the difference is that there is memory attached to the machine and transitions are not labelled with simple inputs but with functions that operate on inputs and memory values.

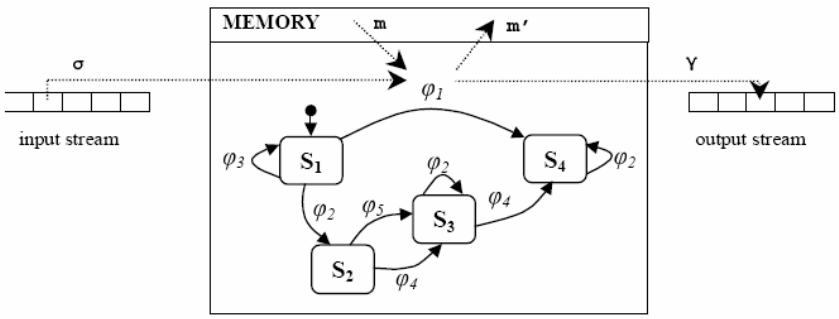

Figure 5: FSA (Finite State Automata) – abstract scheme [30]

We will not cover agents modelling any more, but one has to have in mind that many different tools and environments have been developed too.

7. SECURITY ISSUES

As we have already said, agents act on user's behalf. Usually, this means that agents are working with data which are confidential and which have to be protected. Also, agent's logic should be hidden from others too [48]. For example, agents are sometimes buying goods and possess credit card numbers or so, and it is obvious that this data should be kept. Very often it is also important to protect host's resources as well as agent's data [29].

When discussing security, who is doing something is of great importance. Namely, mobile agent can travel trough the network and come to some place where he is not welcome. Each agent could be a virus or a spy and could be sent by anyone. Also, execution of their actions could do harm to the host.

An agent may also die; in that case it is hard to find out what an agent did in the past and what jobs he carried out.

According to some authors agent should be built using KISS (Keep It Simple, Stupid) principle. In that way agent doesn't know much and the possibility to steel some data an agent possesses is reduced.

Some authors think that messages exchanged by some agent should be ciphered so others couldn't need them. Authorization mechanism should be implemented too; some agents could provide some services only to agent which were authorised.

Based upon Asimov's laws of robotics authors in [53] have tried to define constraints which stop agents to harm something or somebody; the main idea was to make agent safe for the environment.

8. EXAMPLES OF USAGE

Although agent technology is not that old, people have come to the conclusion that we could benefit from combining agent technology with some other technologies. During the years agents and MASs were used in many different fields for solving different kinds of problems. Let us look at some examples.

Today it is difficult to buy some good; we have more information available then ever, and internet-base commerce has helped this process [39]. One can visit many stores online, so the difference is that you don't have to move or leave your current position. The critical task is how to collect and process all this information; Internet agents (assistants) have been designed in order to collect and filter information from the network [**Error! Reference source not found.**9].

In [47] a model where each student sends his personal agent to the virtual university was proposed; this agent can perform different tasks like authentication, information collection, sending the results to some available teacher and so on.

Nowadays it is also very hard and time-consuming to find certain information on the Internet due to its size and speed (problems with network connections, etc.), so any help is more then welcome. In [36] information agents which search the WEB according to user's desires were presented.

Large investments are usually a long-life projects whose outcome is hard to predict, but they are of great strategic importance [10]. Some difficulties may arise during the years which are hard to predict (disasters, war, "11.9" etc.). On the other hand these projects can be very profitable so they require planning and constantly

monitoring and reviewing of relevant information which is hard to; projects information can change very often due to the dynamic environment. Classical DSS can not support all the information needed, so new solutions are presented where emerging computing technologies are used to provide a better support. In [10] authors have presented an idea of how intelligent agents can be used in the context of large investments management.

In [31] authors discuss how to use machine learning techniques and agent technology; an example has been built where agents use machine learning technique in order to choose optimal actions in a robotic football game. A model has been built and agent can learn from its experience and in that way significantly improve the overall performances.

A series of discoveries that suggest new possibilities for life in space made it reasonable to consider the virtual presence in space [41]. Remote agent architecture based on principles of model-based programming, on-board deduction and search, and goal-directed commanding, which integrates constraint-based planning and scheduling, multi-thread execution and reconfiguration has been proposed in order to assist in establishing a virtual presence in space [41].

According to [13] logic programming and MAS can be used for rapid prototyping of new software products. Authors stressed that although software products are being developed for over 30 years, there is always a question whether the product satisfies the criteria of correctness and reliability; this is especially noticeable in distributed systems where entities have to cooperate and communicate with different sources.

In [51] it is described how to *agentify* an information system (IS); certain components have to be added in order to represent memory, commitments and claims. IS has to be able to represent information about current state (this is done using a database), check and process incoming messages, to keep relevant data in memory for some period of time, to communicate with other agents and so on [51].

In [3] authors have tried to show that active databases and agents rely upon similar concepts and that there is a possibility to use and combine results from both fields; that's why the functionality, structure and purpose of both fields have been described and compared. In [42] authors went one step further; they have represented the knowledge of an agent by means of ECA rules optimized using a genetic algorithm. In [23] authors have described and delineated lessons learned from using ECA rules as a means for an implementation of a task shared cooperation protocol.

In [2] MAS was used for meteorological purposes; different air attributes are measured and validated by set of agents, and data mining technique has been used for making these agents intelligent. In [49] authors described how to make benefits from using agent technology for data mining and analysis techniques. In [9] and [18] agents are used to buy and sell some goods. In [17] client agents are used for active caching mechanism in order to provide fast access to the desired information. In [43] networks which can adapt autonomously to dynamic environments by means of agents which are placed on the network nodes are presented. In [27] agent architecture for DS applications in general has been presented as well as for medical care in particular. In [40] MAS has been

developed which helps to organize transplant medical team when an organ which has to be transplanted arrives into the hospital. The goal was to avoid delays and mistakes in organization.

Looking at these chosen examples, you can see that agents are able to solve different kinds of rather complex tasks due to their properties, communication, cooperation and coordination possibilities. More interested readers are referred to [20, 38, 55].

9. CONCLUSION

Taking into consideration that agents were and still are interesting field of study in the last decade, we have tried to give an overview of what has been done in the field in recent years. Many different definitions and properties describing agents were listed. We have also looked at agent architectures which have emerged and showed some examples of their usage. Usually one agent is not enough to solve some problem; that is the reason many agents are usually used. Generally speaking, MAS provides better results: parallel processing increases the speed and improves performances, system stability is increased too, flexibility and reusability are just some of the merits. These good characteristics make them suitable to resolve problems in many different fields.

We have shown how agents are modelled and we have also discussed security issues. One of the really big problems concerning agent systems is security; an agent could be malicious or even infatuated etc. Although one can find many different examples of agent's usage, their local knowledge, lack of security and system control (global constraints) hinder their wider usage. Since this paper was intended to give a slight overview, we hope it represents an interesting source of information for anyone interested in the field.

REFERENCES:

[1] Agents, <http://jude.sourceforge.net/jude/node5.html>

[2] Athanasiadis, I.N., P.A. Mitkas: An agent-based intelligent environmental monitoring system, Management of Environmental Quality: An International Journal Vol. 15, pp. 238-249, 2004.

[3] Bailey, J., M. Georgeff, D. Kemp, D. Kinny, K. Ramamohanarao: Active databases and agent systems – a comparison, Proceedings of the second international workshop on rules in database systems, Lecture notes in computer science 985, pages 342-356, Athens, Greece, 1995.

[4] Beale, R., A. Wood: Agent-Based Interaction, Proceedings of HCI'94, Glasgow, UK, pp. 239-245, 1994.

[5] Berndtson, M., S. Chakravarthy, B. Lings: Coordination Among Agents Using Reactive Rules, pristupano 26.07.2004., <http://64.233.183.104/search?q=cache:AwtECoPL70YJ:www.ida.his.se/ida/research/tech_reports/reports/tr96/HS-IDA-TR-96-011.ps+Coordination+Among+Agents+Using+Reactive+Rules&hl=hr>, 1996.

[6] Bornscheuer, S.E.: Integrating Reactive and Reflective Reasoning by generating Rational Models, Lecture Notes In Computer Science; vol. 1502, Selected papers

from the 11th Australian Joint Conference on Artificial Intelligence on Advanced Topics in Artificial Intelligence, pp. 83-94, 1998.

[7] Bounabat, B., R. Romadi, S. Labhalla: Designing Multi-Agent Reactive Systems: A Specification Method Based on Reactive Decisional Agents, H. Nakashima, C. Zhang (Eds.): PRIMA'99, LNAI 1733, pp. 197-210, Springer-Verlag Berlin Heidelberg, 1999.

[8] Caglayan, A., C. Harrison: Agent sourcebook, John Wiley & Sons, New York, 1997.

[9] Chavez, A., P. Maes: Kasbah: An Agent Marketplace for Buying and Selling Goods, <http://citeseer.ist.psu.edu/cache/papers/cs/22903/http:zSzzSzwww-ec.njit.eduzSz~bartelzSzNegoPapzSzKasbahMIT.pdf/chavez96kasbah.pdf>, 1996.

[10] Collan, M., S. Liu: Fuzzy logic and intelligent agents: towards the next step of capital budgeting decision support, Industrial Management & Data Systems, vol. 103, pp. 410-422, 2003.

[11] Cordenonsi, A.Z., L.O. Alvares: An Evolutionary Behavior Tool for Reactive Multi-agent Systems, G. Bittencourt and G. Ramalho (Eds.): SBIA 2002, LNAI 2507, pp. 334-344, Springer-Verlag Berlin Heidelberg, 2002.

[12] D'Inverno, M., D. Kinny, M. Luck, M. Wooldridge: A Formal Specification of dMARS, In Intelligent Agents IV: Proceedings of the Fourth International Workshop on Agent Theories, Architectures and Languages, Singh, Rao and Wooldridge (eds.), Lecture Notes in AI, 1365, Springer-Verlag, 1998.

[13] Dart, P., E. Kazmierczak, M. Martelli, V. Mascardi, L. Sterling, V. S. Subrahmanian, F. Zini: Combining Logical Agents with Rapid Prototyping for Engineering Distributed Applications, <http://citeseer.ist.psu.edu/cache/papers/cs/5547/ftp:zSzzSzftp.disi.unige.itzSzpersonzSzZiniFzSzPaperszSzstep99.pdf/combining-logical-agents-with.pdf>, 1999.

[14] Davis, D., A. Sloman, R. Poli: Simulating agents and their environments, <http://www2.dcs.hull.ac.uk/NEAT/dnd/papers/aisbq.pdf >, 1995.

[15] Dell'Acqua, P., M. Engberg, L.M. Pereira: An Architecture for a Rational Reactive Agent, <http://centria.di.fct.unl.pt/~lmp/publications/online-papers/epia03-agent.pdf>, 2003.

[16] D'Inverno, M., M. Luck: Understanding agent systems, Springer, Berlin Heidelberg 2001.

[17] Erni, A., M. C. Norrie: Agent Based Internet Database Services, <http://citeseer.ist.psu.edu/cache/papers/cs/1263/ftp:zSzzSzftp.inf.ethz.chzSzpubzSzpublicationszSzpaperszSziszSzglobiszSz1997g-en-doccaise.pdf/erni97agent.pdf >, 1997.

[18] Eymann, T.: Digitale Geschäftsagenten, Springer-Verlag Berlin Heidelberg, 2003.

[19] Feijó, B., P.C. Rodacki Gomes, J. Bento, S. Scheer, R. Cerqueira: Distributed agents supporting event-driven design processes, J. S. Gero and F. Sudweeks (eds), Artificial Intelligence in design, Kluwer Academic Publishers, pp. 557-577, 1998.

[20] Ferber, J.: Multiagenten systeme, Addison Wesley, New York, 2001.

[21] Gabbay, D., R. Nossum, M. Thielscher: Agents in Proactive Environments, <http://www.doc.ic.ac.uk/research/technicalreports/1997/DTR97-5.pdf>, 1997.

[22] Geppert, A., M. Kradolfer, D.Tombros: Realization of Cooperative Agents Using an Active Object-Oriented Database Management System, Proc. 2nd Workshop on Rules in Databases (RIDS), Athens, Greece, 1995.

[23] Hagen, I., M.B., S. Chakravarthy, B. Lings: Challenges for ECA Rules Designers when Implementing Coordination Protocols, < http://citeseer.ist.psu.edu/cache/papers/cs/830/http:zSzzSzwww.ida.his.sezSzidazSzresearchzSztech_reportszSzreportszSztr98zSzHS-IDA-TR-98-006.pdf/challenges-for-eca-rule.pdf> , 1998.

[24] Hayes-Roth, B.: An architecture for adaptive intelligent systems, Artifcial Intelligence Vol. 72, 1995.

[25] Hexmoor, H.: Evolution of Agent Architectures, W. Truszkowski, C. Rouff, M. Hinchey (Eds.): WRAC 2002, LNAI 2564, pp. 469-470, Springer-Verlag Berlin Heidelberg, 2003.

[26] Hindriks, K.V., F.S. de Boer, W. van der Hoek, J.-J. Meyer: A programming logic for part of the agent language 3APL, Lecture Notes In Computer Science; Vol. 1871, Proceedings of the First International Workshop on Formal Approaches to Agent-Based Systems-Revised Papers, pp. 78-89, 2000.

[27] Huang, J., N. R. Jennings, J. Fox: An Agent Architecture for Distributed Medical Care, <http://citeseer.ist.psu.edu/cache/papers/cs/987/ftp:zSzzSzftp.elec.qmw.ac.ukzSzpubzSzisagzSzdistributed-aizSzpublicationszSzIntelligent-Agents95.pdf/huang95agent.pdf>, 1995.

[28] Jung, C.G., K. Fischer: Logic-Based Hybrid Agents, A.C. Kakas, F. Sadri (Eds.): Computational Logic (Kowalski Festschrift), LNAI 2407, pp. 626-654, Springer-Verlag Berlin Heidelberg, 2002.

[29] Karnik, N.M., A.R. Tripathi: Security in the Ajanta Mobile Agent System, pristupano 10.09.2004, <http://citeseer.ist.psu.edu/cache/papers/cs/8605/http:zSzzSzwww.cs.umn.eduzSzAjantazSzpaperszSzsecurityPaper.pdf/karnik99security.pdf >, 1999.

[30] Kefalas, P.: Formal Modelling of Reactive Agents as an Aggregation of Simple Behaviours, I.P. Vlahavas and C.D. Spyropoulos (Eds.): SETN 2002, LNAI 2308, pp. 461-472, Springer-Verlag Berlin Heidelberg, 2002.

[31] Kostiadis, K., H. Hu: Reinforcement Learning and Co-operation in a Simulated Multi-agent System, <http://citeseer.ist.psu.edu/cache/papers/cs/13113/http:zSzzSzprivatewww.essex.ac.ukzSz~kkostizSzFileszSzIROS99.pdf/kostiadis99reinforcement.pdf>, 1999.

[32] Kowalski, R., F. Sadri: An Agent Architecture that Unifies Rationality with Reactivity, http://citeseer.ist.psu.edu/cache/papers/cs/15910/http:zSzzSzwww-lp.doc.ic.ac.ukzSzUserPageszSzstaffzSzrakzSzpaperszSzagents-97.pdf/kowalski97agent.pdf, 1997.

[33] Kowalski, R., F. Sadri: Toward a unified agent architecture that combines rationality with reactivity, <http://citeseer.ist.psu.edu/cache/papers/cs/3592/http:zSzzSzwww-lp.doc.ic.ac.ukzSz_lpzSzSadrizSzunify.pdf/kowalski96towards.pdf>, 1996

[34] Lanzi. P. L.: Adaptive Agents with Reinforcement Learning and Internal memory, Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior, MIT Press, 2000.

[35] Lingnau, A., O. Drobnik, P. Dömel: An HTTP-based Infrastructure for Mobile Agents, http://www.w3.org/Conferences/WWW4/Papers/150/, 1995.

[36] Loke, S.W., L. Sterling, L. Sonenberg: A knowledge-based approach to domain specialized information agents, Internet Research: Electronic Networking Applications and Policy, vol. 9, pp. 140-152, 1999.

[37] Maleković, M., M. Čubrilo: Knowledge, Belief, And Desire In Multi-Agent Systems. CD Proceedings, The 7th IEEE International Conference on Intelligent Engineering Systems, INES 2003.

[38] Marik, V., O. Stepankova, H. Krautwurmova, M. Luck.: Multi agent systems and aplications II, Springer-Verlag Berlin Heidelberg, 2002.

[39] Martins, R.M., M.R. Chaves, L. Pirmez, L.F. Rust da Costa Carmo: Mobile agents applications, Electronic Networking Applications and Policy, vol. 11, pp. 49-54, 2001.

[40] Moreno, A., A. Valls, J. Bocio: Management of Hospital Teams for Organ Transplants Using Multi-agent Systems, S. Quaglini, P. Barahona, and S. Andreassen (Eds.): AIME 2001, LNAI 2101, pp. 374-383, Springer-Verlag Berlin Heidelberg, 2001.

[41] Muscettolay, N., P.P. Nayakz, B. Pellz, B.C. Williams: Remote Agent: To Boldly Go Where No AI System Has Gone Before, <http://mers.csail.mit.edu/papers/aij98.pdf>, 1998.

[42] Nonas, E., A. Poulovassilis: Optimisation of Active Rule Agents Using a Genetic Algorithm Approach, <http://www.dcs.bbk.ac.uk/~ap/pubs/dexa98.pdf>, 1998.

[43] Nonas, E., A. Poulovassilis: Optimising Self Adaptive Networks by Evolving Rule-Based Agents, <http://www.dcs.bbk.ac.uk/~ap/pubs/evo99.pdf>, 1999.

[44] Qureshi, F., D. Terzopoulos, R. Gillett: The Cognitive Controller: A Hybrid, Deliberative/Reactive Control Architecture for Autonomous Robots, R. Orchard et al. (Eds.): IEA/AIE 2004, LNAI 3029, pp. 1102-1111, Springer-Verlag Berlin Heidelberg, 2004.

[45] Sackmann, S.: Bilaterale Preisverhandlungen von Software-Agenten, Deutscher Universitäts-Verlag, Wiesbaden, 2003.

[46] Silva, A., M.M. da Silva, J. Delgado: An Overview of AgentSpace: A Next-Generation Mobile Agent System, <http://berlin.inesc.pt/agentspace/main-eng.html>, 1998.

[47] Stenge, I., U. Bleimann, J. Stynes: Social insects and mobile agents in a virtual university, Campus-Wide Information Systems, vol. 20, pp. 84-89, 2003.

[48] Subrahmanian, V. S. and others: Heterogeneous Agent systems, Mit Press, USA, 2000.

[49] Sugumaran, V., R. Bose: Data analysis and mining environment: a distributed intelligent agent technology application, Industrial Management & Data Systems Vol. 99 No. 2., 1999.

[50] Tessier, C., L. Chaudron, H. Müller: Conflicting Agents, Kluwer Academic Publishers, Boston, 2001.

[51] Wagner, G.: Towards Agent-Oriented Information Systems, <http://www.inf.fu-berlin.de/_wagnerg/AOIS.ps>, 2000.

[52] Wagner, G.: Vivid Agents: How They Deliberate, How they React, How They Are Verified, <http://lips.informatik.uni-leipzig.de/pub/showDoc.Fulltext/dokument.pdf?lang=en&doc=1996-31&format=pdf&compression=&rank=0>, 1996.

[53] Weld, D., O. Etzioni: The first law of Robotics (a call to arms), <http://www.cs.washington.edu/homes/weld/papers/first-law-aaai94.pdf>, 1994.

[54] What is KQML?, <http://www.cs.umbc.edu/kqml/whats-kqml.html>

[55] Wooldridge, M., N. R. Jennings: Agent Technology Foundations, Applications, and Markets, Springer, Berlin, 2002.

[56] Wooldridge, M., N.R. Jennings: Intelligent agents: theory and practice, The Knowledge Engineering Review, vol. 10, pp. 115-152, 1995.

Received: 20 December 2005
Accepted: 26 June 2006