TOWARDS AN AGENT-BASED TUTORING ARCHITECTURE

I. D. Zaharakis[1]
john@math.upatras.gr

A. D. Kameas[2]
kameas@math.upatras.gr

P. E. Pintelas[1]
pintelas@math.upatras.gr

ABSTRACT

This paper proposes a multi-layer agent-based architecture for Intelligent Tutoring Systems (ITS). It specifies how the agents reason about by using attitudes such as beliefs (B), desires (D) and intentions (I) and adopts the formal methods of BDI logic. In an attempt to bridge the gap between agent theories and real agent systems, it proposes an algorithm of agent behavior in the multi-agent environment and applies this algorithm in a real tutoring session.

INTRODUCTION

As the term *agent* is used widely these days, the problem of the agent definition is raised. Several attempts have been made for the definition of the agent (in the concept of Artificial Intelligence). Marvin Minsky says [18]: "In the Society of Mind [19], the idea was to use the word agent when you want to refer to a machine that accomplishes something", N. Shardlow suggests "Agents do things, they *act*: that is why they are called agents" [27], while other researchers use notions to describe an agent that are normally applied to humans, such as knowledge, beliefs, desires, intentions, capabilities, choices, commitments or obligation, [28, 6, 23] or even emotions [2]. Wooldridge and Jennings in [30] use a broader definition and argue that an agent denotes a hardware or software-based system that exhibits properties such as autonomy, social ability, reactivity, and pro-activeness. Several other agent properties have been suggested by other researchers. These include mobility, veracity, benevolence and rationality.

In our work, we consider an agent as an *intentional system* based on the description of the philosopher Daniel Dennett, who, by this term means entities "whose behavior can be predicted by the method of attributing belief, desires and rational acumen" [7]. The occasions where the *intentional stance* is appropriate have been discussed by McCarthy [17], and later by Seel [26] and Rosenschein and

---

[1] Division of Computational Mathematics & Informatics and Educational Software Development Laboratory, Department of Mathematics, University of Patras, Hellas.
[2] Educational Software Development Laboratory, Department of Mathematics, University of Patras, Hellas.

Kaelbling [24] showed that more or less every object can be described by the intentional stance. In addition to the fact that an intentional system seems to be a *necessary* condition for agenthood, it is a *sufficient* condition, too, as it is pointed out by Shardlow in [27]. Concluding that an agent is a system that is most conveniently described by the intentional stance, Wooldridge and Jennings [30] consider that the appropriate attitudes for representing agents are classified in *information attitudes* (belief and knowledge) which are related to the information that an agent has about the world it occupies, and *pro-attitudes* (desire, intention, obligation, commitment, choice etc.) that in some way guide the agents' actions.

For the conceptualization of an *agent* (and more generally, of a *multi-agent system*), in this paper the logical framework developed by Rao and Georgeff in [23] is adopted and their formal methods for representing and reason about agents' properties are used. According to these, a multi agent system is viewed as having the three mental attitudes of belief, desire and intention (BDI). Beliefs are the information (which may be incomplete or incorrect) the agent has about the world he acts upon and the information he sends to the world; they represent an agent' s *informational* state. Desires are the tasks or the objectives that the agent is allocated or has to accomplish and represent the agent' s *motivational* state. Intentions are the agent' s commitments to a desire and represent his *deliberation* state. In addition, every agent contains a *plan library*. Plans are the possible ways that an agent can bring about an intention. In general, a plan is a partial commitment on how to achieve a desire. The BDI formalism has been chosen as the underline formalism of our architecture because it captures the intentional stance and it has an associated mathematical theory with sound and complete axiomatizations and methods for the satisfiability and validity of formulas; such axiomatizations and methods are discussed in [23].

This paper is part of an ongoing research effort with threefold objective:

(*i*) to propose and formally substantiate an agent-based architecture of an ITS that encompasses mental attitudes such as beliefs, desires and intentions,
(*ii*) to describe how the agents in a multi-agent tutoring environment reason, act and negotiate, and
(*iii*) to contribute in bridging the gap between agent theories and real agent application.

The design and development of an ITS-generator called X-GENITOR has been selected as an implementation testbed.

Thus, after presenting in section 2 the formal methods and properties of the BDI formalism and discussing other agent-based systems, the redesign of GENITOR, an existing ITS-generator, into X-GENITOR, an agent-based system, is described in section 3, together with an algorithm of agent behavior and a brief example. The paper concludes with a summary of the work and future research of the authors.

BACKGROUND

1. BDI Formalism

The notion of BDI agent is inspired from the philosophical theories of Bratman [3], who argues for the significance of the intention attitude in practical reasoning and the reasons why it is not reducible into beliefs and desires. A BDI logic is a multi modal temporal logic based on the branching time logic CTL* [8]. The basic concept of the BDI logic is the transformation of a decision tree and the functions applied to it, to an equivalent model that represents beliefs, desires and intentions as accessibility relations over a set of possible worlds. Each world is a time tree with a single past and a branching future, with nodes corresponding to a possible system state and representing the options available to the system itself or the state of the environment which the system is embedded in, and with arcs (transitions) representing the different system actions for the achievement of an objective.

In this way, there are *belief-, desire-* and *intention-accessible worlds* corresponding to the states that the system believes to be possible, desires to bring about and intends to bring about, and *belief-, desire-* and *intention-accessibility relations* ($\mathcal{B}$, $\mathcal{D}$ and I, respectively) which are the transitions between the worlds. The syntax and semantics of the BDI logic which is adopted in the proposed multi agent architecture are described in [23]; in the following only its basic properties are briefly presented.

The normal modal system KD45 (weak-S5) is adopted for beliefs, while the K and D axioms are adopted for desires and intentions. K-axiom states that if an agent believes (desires or intends) p and believes (desires or intends) that $p \supset q$ then he will believe (desire or intend) q D-axiom says that the agent' s beliefs (desires or intentions) are not contradictory. 4-axiom and 5-axiom are respectively the positive and negative introspection axioms: 4-axiom states that an agent is aware of what it knows and 5-axiom says that an agent is aware of what it does not know. The axioms of the KD45 system are summarized in Figure 1, where p, q are propositions of the classical propositional logic, "$\wedge$"and "$\supset$" are the propositional connectives for conjunction and implication, and BEL, DES and INTEND are modal operators representing the agent's beliefs, desires and intentions respectively.

	Beliefs	Desires	Intentions
K-axiom	$\text{BEL}(p) \wedge \text{BEL}(p \supset q) \supset \text{BEL}(q)$	$\text{DES}(p) \wedge \text{DES}(p \supset q) \supset \text{DES}(q)$	$\text{INTEND}(p) \wedge \text{INTEND}(p \supset q) \supset \text{INTEND}(q)$
D-axiom	$\text{BEL}(p) \supset \neg\text{BEL}(\neg p)$	$\text{DES}(p) \supset \neg\text{DES}(\neg p)$	$\text{INTEND}(p) \supset \neg\text{INTEND}(\neg p)$
4-axiom	$\text{BEL}(p) \supset \text{BEL}(\text{BEL}(p))$		
5-axiom	$\neg\text{BEL}(p) \supset \text{BEL}(\neg\text{BEL}(p))$		

Figure 1: The axioms of the KD45 system.

The necessitation rule is applied in beliefs, desires and intentions (Figure 2)[3]:

	Necessitation rule
Beliefs	If $\vdash p$ then $\vdash \mathrm{BEL}(p)$
Desires	If $\vdash p$ then $\vdash \mathrm{DES}(p)$
Intentions	If $\vdash p$ then $\vdash \mathrm{INTEND}(p)$

Figure 2: The necessitation rule for beliefs, desires and intentions.

As the belief-, desire- and intention-accessible worlds are sets and in the same time are time trees, there are set relationships ($\subseteq$, $\cap$ etc.) and structure relationships (sub-world, identical or incomparable worlds) between them. Depending on the combinations of the above relationships and the axiomatizations of them, three constraints are under consideration: *strong realism* [21], *realism* [6] and *weak realism* [22]. We will describe some of the characteristics of the weak realism, as it is the adopted constraint for our agents; the reader may refer to [21], [6] for the other two constraints. The intersection between the belief-accessible worlds and desire-accessible worlds is not null and an agent does not desire a proposition the negation of which is believed. The same holds between desire- and intention-accessible worlds and between belief- and intention-accessible worlds. An agent does not intend a proposition the negation of which is desired or believed (Figure 3).

Semantic Condition	Distinguishing Axiom
$\mathcal{B} \cap \mathcal{D} \neq \varnothing$	$\mathrm{BEL}(p) \supset \neg\mathrm{DES}(\neg p)$
$\mathcal{D} \cap I \neq \varnothing$	$\mathrm{DES}(p) \supset \neg\mathrm{INTEND}(\neg p)$
$\mathcal{B} \cap I \neq \varnothing$	$\mathrm{BEL}(p) \supset \neg\mathrm{INTEND}(\neg p)$

Figure 3: The adopted BDI modal system.

Some other properties that are satisfied by the adopted model are the *asymmetry thesis* proposed by Bratman [3] and extended by Rao and Georgeff [22] and the *consequential closure principles*. According to the asymmetry thesis, intention-belief, intention-desire and desire-belief inconsistency is not allowed, whereas intention-belief, belief-intention, intention-desire, desire-intention, desire-belief and belief-desire incompleteness is allowed (Figure 4)[4].

---

[3] '$\vdash$' means that the formula which follows is valid.

[4] '$\models$' is the satisfaction relation and means that the formula which follows is satisfiable.

	Asymmetry Thesis
intention-belief consistency	$\models \text{INTEND}(p) \supset \neg\, \text{BEL}(\neg p)$
intention-desire consistency	$\models \text{INTEND}(p) \supset \neg\, \text{DES}(\neg p)$
desire-belief consistency	$\models \text{DES}(p) \supset \neg\, \text{BEL}(\neg p)$
intention-belief incompleteness	$\not\models \text{INTEND}(p) \supset \text{BEL}(p)$
belief-intention incompleteness	$\not\models \text{BEL}(p) \supset \text{INTEND}(p)$
intention-desire incompleteness	$\not\models \text{INTEND}(p) \supset \text{DES}(p)$
desire-intention incompleteness	$\not\models \text{DES}(p) \supset \text{INTEND}(p)$
desire-belief incompleteness	$\not\models \text{DES}(p) \supset \text{BEL}(p)$
belief-desire incompleteness	$\not\models \text{BEL}(p) \supset \text{DES}(p)$

Figure 4: Asymmetry thesis principles.

Although, consequential closure is referred in the literature [1,6] as a problem[5], Rao and Georgeff argue that it can be seen as a property that needs to be satisfied [23]: "... it is rational for an agent to intend p and at the same time not intend q, no matter how strong the belief about $p \supset q$". The so-called belief-intention consequential closure principle is required between intentions and desires, and between desires and beliefs, too.

2. Architectural Approaches

Several agent-based architectures have already been proposed in the literature. For example, *Procedural Reasoning System* (PRS) [12], one of the first implemented systems based on a BDI architecture, contains a plan library and explicitly represents beliefs, desires and intentions. Beliefs are facts, expressed in first-order logic, and represent the state of the environment or the internal state of the system. Desires are dynamic representations of the *system behavior*. Plans, called *knowledge areas*, are associated with an *invocation condition* and may be *activate* or *reactive*. The active knowledge areas are the system's intentions. All those date structures are manipulated by the *system interpreter*. PRS and its successor dMARS have been evaluated in a number of large scale applications such as maintenance procedures of the space shuttle, telecommunications network management, air-combat modeling and business process management. *Intelligent Resource-bounded Machine Architecture* (IRMA) [4], like PRS, is an implemented BDI architecture and includes a plan library and data structures for representing beliefs, desires and intentions. In addition, it contains a *reasoner*, a *means-end analyser,* an *opportunity analyser*, a *filtering process* and a *deliberation process* which are responsible for

[5] Intuitively, it states that if an agent intends to do an action he has to intend all the side-effects of such an action.

reasoning, choosing options, determining intentions and actions, and observing the world. IRMA has been evaluated in the *Tileworld* [20] experimental scenario. On the other hand, GRATE* [15], is a layered architecture in which the agents are characterized by the mental attitudes of beliefs, desires, intentions and joint intentions, and consists of a domain level system which is responsible for the organization, a cooperation layer which interfaces to the domain level system and specifies the agents' actions for cooperative problem solving, and a control layer responsible for the domain level system's activities to be coordinated with those of the other agents of the environment. GRATE* has been evaluated in electricity transportation management. Although, the above systems have been evaluated in a broad range of domains, it has not proven that these approaches are applicable to the special requirements of ITS. Furthermore, the agent-based ITS architectures that have been developed focus primarily on inter-agent communication and not on the representation of the ITS modules. For example, GIA [5] adopts a *federated architecture* [11], with a set of *facilitators* and a set of *kernel* and *interface agents*, and supports both standalone and networked configurations. The principal research task of the GIA project lies in the design of an expressive formal language and in the identification of the properties of such a language for the demands of an ITS. FITS [14], is a platform independent shell and contains agents for student and learner modeling. Like GIA, FITS is concerned primarily on communication between agents.

In this paper, an agent-based architecture for GENITOR, an ITS generator is described. Consequently, the modeling scope is far broader than the approaches described above, in that it has to meet design requirements such as massive agent generation, agent communication and synchronization, specification of a large number of agents with diverse objectives by non-computer experts etc. The adopted BDI formalism sets a well-defined formal framework that does not constraint the flexibility and variability inherent in a tutoring session. Nevertheless, existing architectures have to be extended in order to meet the aforementioned special requirements.

THE PROPOSAL

1. The Tutoring Architecture of GENITOR

GENITOR [16] is an ITS generator that can be used for the development of autonomous intelligent training applications in diverse subjects [13], [10]. Such an application attempts to transfer to the trainees two kinds of knowledge: procedural knowledge on how to apply a methodology and declarative knowledge to support the application of the methodology.

The declarative knowledge in a GENITOR application consists of Application Learning Units (ALU). ALUs are hypermedia constructs of monomedia Learning Units (LU) and make up the domain base of the application. ALUs, are described by attributes that represent both static information used to identify the unit (i.e. title,

type, location, size etc.) and dynamic information that describes the behavior of the unit (i.e. pedagogical prerequisites and objectives, display constraints etc.).

The learning scenario embedded in an application developed with GENITOR is made up of stages. A stage is an integral application module and implements a certain instructional strategy as part of the learning scenario. It contains its own domain base (which is a subset of the application domain base) and manages its own tutoring interface, offering tutoring actions to students and attempting to satisfy the tutoring goal hierarchy described in the instructional strategy specified by the application authors.

In order to develop an ITS, authors have to use the tools of GENITOR in order to describe the methodology to be taught (procedural knowledge base) and develop the LUs and ALUs of the declarative knowledge base. Two expert systems, MeT [32] and DES, assist authors during this process, while they guide domain presentation during ITS execution. Actually, in the latest version of GENITOR, MeT has been replaced with HMeT [33], a hybrid expert system that uses three agents during the tutorin process. The authors have to describe the learning cycle by initializing appropriately the system-offered stage templates. The system offers a consistent user interface for all its tools, and is also equipped with an application prototyping facility.

2. The Proposed New System Architecture

The new architecture of X-GENITOR will be populated by autonomous, self-reactive agents who are knowledgeable of the capabilities of each other and are able to collaborate by forming teams. As a result, they will be able to accomplish complex tasks or tasks which cannot be performed by one agent itself. These agents, who have evolved from the GENITOR ALUs concept, will have their own user interface and make use of multiple media formats, while several instructional strategies and user models are represented in their mental states.

All these require the development of formalisms and algorithms for team formation and team action, together with protocols for communication and mechanisms for handling the user interfaces. In order to better estimate the feasibility of the approach, the complexity of the process and the effort required, a first prototype of X-GENITOR has been built around a hierarchical multi-layer architecture (Figure 5), which constitutes a simplification of general deliberative agent architectures with the following assumptions:

- The agent hierarchy consists of two levels, top and bottom.
- There exists only one agent (*Pedagogy Agent*) at the top level, who has knowledge of all other agents and the tasks they can accomplish, and can refine a composite goal into simpler tasks.
- All other agents (*Application Learning Unit Agents* (*ALU-Agents*)), who reside at the bottom hierarchy level, have knowledge of (a) the top level agent, and (b) the tasks they can accomplish.

- A point-to-point communication protocol between the top level agent and the ALU-Agents has been adopted; no direct communication takes place among the ALU-Agents.
- For implementation purposes, a *User Interface Agent* has been introduced, who is responsible for the interaction between the trainee and the system; this agent is not part of the hierarchy.

The modeling of the trainee and of the user interface agent falls outside the scope of this paper. The trainee is seen only as an event generator whose events affect the agents' mental state. The User Interface Agent which can be modeled with a BDI logic, handles the user events, determines the running procedures in the user interface and sends requests to the Pedagogy Agent. If the latter can fulfill these requests then he responds to the user interface agent, otherwise he passes the requests to one or more ALU-Agents.

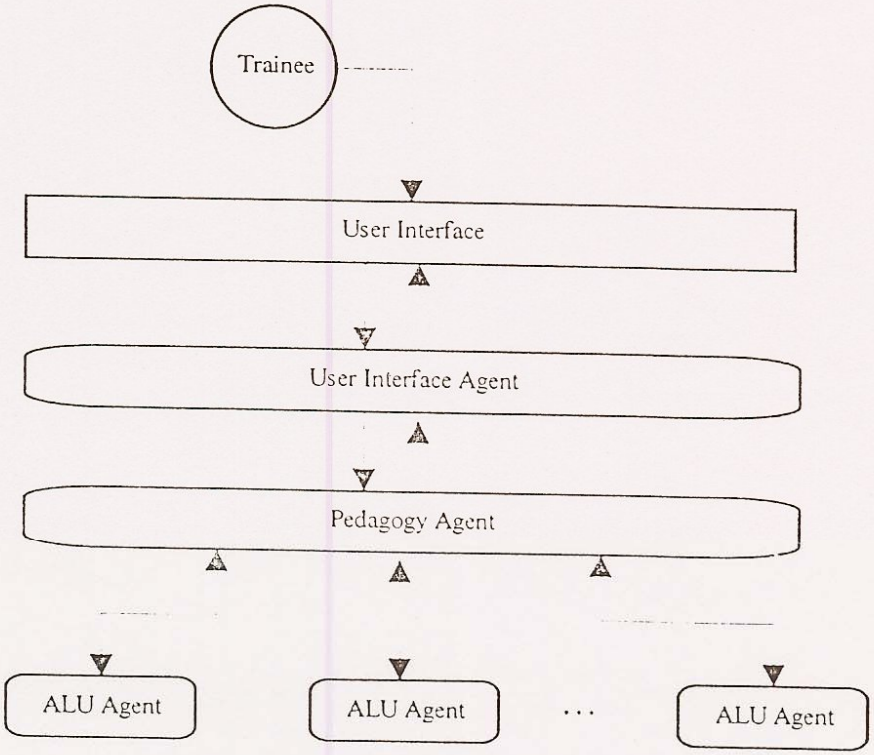

Figure 5: System architecture.

The Pedagogy Agent can access several instructional strategies and acts according to these. He also knows which of the ALU-Agents have the skills to accomplish a certain task contained in an instructional strategy. Because of this knowledge, he can adopt intentions about a task which he knows that can be accomplished by some ALU-Agents. In fact, he can form teams of ALU-Agents. ALU-

Agents contain elementary course parts about the teaching subject. An elementary course part can be exhibited in several different ways; this depends on the ALU-Agent skills, the instructional strategy and the user model. The latter does not comprise a separate agent or component. Instead, it is implicitly specified in the precondition of the ALU-Agents plan. An ALU-Agent has no knowledge of the other ALU-Agents and the only thing it knows is how to bring about his intentions. This means that when a team formation takes place for the accomplishment of a higher goal none of the team members knows the existence of the others except the Pedagogy Agent, which knows all about team members and is the one who coordinates the team formation and activity.

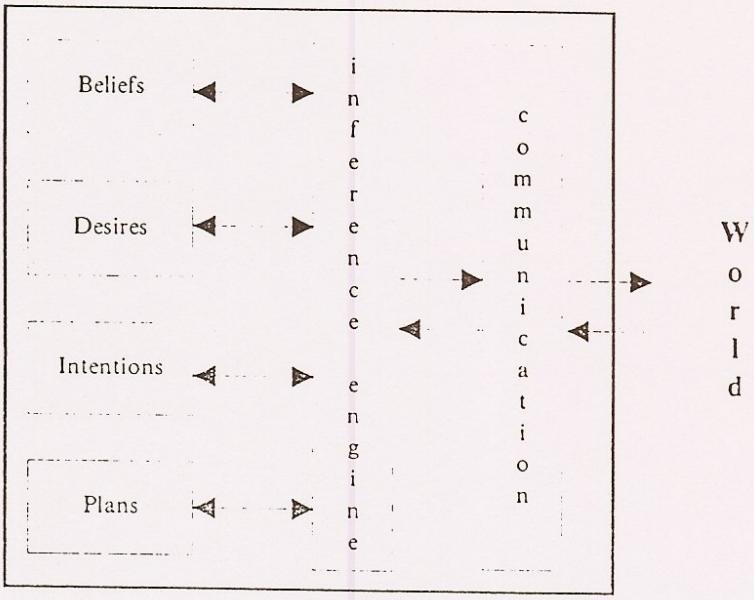

Figure 6: Agent architecture.

The proposed architecture (Figure 6) is a *deliberative* one, i.e. it contains an explicitly represented symbolic model of the world and the decisions are made via logical reasoning based on pattern matching and symbolic manipulation. The close relation between symbolic processing systems and mathematical logic made this choice desirable in the sense that a BDI architecture can be easily implemented in conjunction with well known AI techniques and established software engineering methodologies.

The mental state of the agent consists of beliefs, desires and intentions as those described earlier. Each of these can be implemented as data structures (i.e. PROLOG-like facts or rules). Plans represent the *procedural knowledge* or *know how* [31] of an agent. They specify the ways that an agent brings about his intentions. Typically, plans contain a *descriptor* part which describes the pre-conditions for plan triggering and a *body* part which describes the actions that will take part in

the plan execution[6].

The Inference engine is the mechanism which handles and updates the agent mental state (beliefs, desires and intentions), selects, executes and rejects plans (according to the mental state), and is responsible for the preservation of the properties of the model. The Communication component transfers the observations of the world as well as the current state of the agent to the world. The Communication component entails a kind of openness, in the sense that different programming languages could be used for the agents implementation but just one communication protocol, such as KQML [9]. However, the proposed system does not contain such heterogeneous agents and the communication component plays the role of the transformer of the internal system protocol to agent's internal language and vice versa. All communication actions performed by an agent are treated in the same way as in the *speech acts theory* [25]: every intended communication action from an agent changes the world in an analogous way a physical action does.

Each agent operates according to the following algorithm. Each single iteration of the loop occurs at regular intervals in the same way as in [28]:

Algorithm

(1) *check for events;*

if there are no events then wait for events

else select an event p

(2) *check mental state for p*

(3) *adopt an intention for p*

(4) *find the plans concerning p and execute the ready to fire one*

(5) *revise mental state*

(6) *go to (1)*

The events that occur in the environment and concern the current agent or the events caused by the adopted intentions are kept in a queue. The agent reacts to the events and checks his mental state to confirm which of them are consistent with his mental attitudes, that is, he checks whether the distinguishing axioms hold (Figure 3). Then, he either adopts an intention to execute the requested action or rejects it. In essence, even in the case of rejection, the agent adopts an intention in order to inform the agent which requested the action, for the rejection. When the agent is committed to bring about p then among the plans concerning p only some of them are ready to fire. Randomly, one of them is executed. If a plan will fail then another one from the ready to fire is selected. After the plan execution, independently of success or failure, the adopted intention is dropped, the agent mental state is updated and the environment is informed by causing an internal event q and adopting an intention about q.

---

[6] Several researchers have developed specifications for plan formations using BDI logic [31], [29].

3. Example

Consider an ITS which teaches the concept of gravitation and an instructional strategy contained in the Pedagogy Agent consisting of the following stages:

1. Stage 1: *"Show the theory from the general to specific"*
2. Stage 2: *"Let the trainee explore the mathematical equations and the general theory"*
3. Stage 3: *"Show examples"*

Consider, also, that the following ALU-Agents are needed[7]:

1. ALU-Agent1: *"Elements of the theory of gravitation"*
2. ALU-Agent2: *"Mathematical description"*
3. ALU-Agent3: *"Acceleration and velocity in a gravitational field"*
4. ALU-Agent4: *"Earth's gravitational field"*
5. ALU-Agent5: *"Example: The experiment at the tower of Pizza"*
6. ALU-Agent6: *"General theory: Fields"*, *"General theory: Forces"*, *"General theory: Moving objects"*, *"General theory: Falling objects"*

When the above instructional strategy is executed, the Interface Agent requests Stage 1 from the Pedagogy Agent which in turn adopts intention *"Show the theory from the general to specific"* and requests from ALU-Agent1, ALU-Agent2, ALU-Agent3, ALU-Agent4 and ALU-Agent6 to accomplish a particular task. For example, he requests from ALU-Agent1 to accomplish the task *"Elements of the theory of gravitation"*, from ALU-Agent6 to accomplish the task *"General theory"* and so on. In particular, ALU-Agent6 decides on his own, according on his plans, which of the course parts should be executed. When all ALU-Agents accomplish their tasks, they send the results to the Pedagogy Agent; he composes the parts and determines the order that they will be presented and informs accordingly the Interface Agent which is now responsible for the course parts exhibition. When the Stage 1 is finished, the Interface Agent requests the next stage from the Pedagogy Agent and an analogous process takes place. Finally, in the Stage 3, the ALU-Agent5 is requested for the task *"Example: The experiment at the tower of Pizza"*. Then, according to the user model, implicitly represented in pre-conditions plans of LUs, he may determine the way that the course part will be presented (e.g. simple text, animation, video etc.). In a more analytical example, stages should be further refined into sub-stages in order to adopt the training session to the particular needs of each student. This would, of course, require a deeper analysis of the subject domain, which in turn would lead to a more densely populated agent society and would require a more complex algorithm for the description of agent behavior. However, due to space limitations, the example is of the minimal necessary complexity in order to demonstrate the basic ideas underlying this paper.

---

[7] The tasks accomplished by each agent are included in double quotations. All agents accomplish one task except ALU-Agent6 who accomplishes four tasks.

There are cases where an agent can not bring about a goal in his own but he knows that some other agents can do so. For example, assume that the User Interface Agent (UIA) requests p from Pedagogy Agent (PA) and PA does not explicitly believe p; instead, he believes $r \wedge s \supset p$, and he also believes that ALU-Agent1 (ALUA1) and ALU-Agent2 (ALUA2) believe r and ALU-Agent3 (ALUA3) believes s. Then PA adopts an intention about p and requests r and s from ALUA1 and ALUA2, respectively, which, in turn, adopt intentions about r and s. Let, ALUA2 accomplishes successfully s and informs PA about this, whereas ALUA1 does not accomplish successfully r and informs PA about the failure. Since PA believes that ALUA3 can, also, adopt intentions about r, he requests r from him. If ALUA3 will accomplish successfully r, then p is accomplished and PA drops his intention about p and informs UIA about this; otherwise he informs UIA about the failure.

CONCLUSIONS

This work considers an agent as an intentional system, specifying its behavior with attitudes such as beliefs, desires and intentions. It adopts the formal methods of BDI logic and proposes a multi-layer agent-based architecture of an ITS and an algorithm of agent behavior in a multi-agent environment. Thus, this work attempts to contribute to the bridging of the gap between the agent theories and real agent applications. However, it is a mid-step between the modular systems and the autonomous, reactive multi-agent systems. Although the simplifications embodied in X-GENITOR weaken the overall system, in the sense that only one agent has a global awareness for the other agents and their skills in the environment, he is the one responsible for the task assignment and he determines the roles inside the team, the results from this attempt were encouraging, leading us to helpful conclusions about the used mental attitudes, the team formation, the negotiation and collaboration as well as the refinement of the used algorithm. Our future research aims to massive agent generation from ITS generators which act in common environment, negotiate and compose teams for task accomplishment without the intervention of others (e.g. Pedagogy Agent). We envisage ALU-Agents which have knowledge of the environment and the capabilities of the other agents, handle their own user interface and contain several instructional strategies and user models in their mental states.

REFERENCES

[1] Allen, J. (1990). Two views of intention: Comments on Bratman and on Cohen and Levesque. In P. R. Cohen, J. Morgan, and M. E. Pollack, editors, *Intentions in Communication.* MIT Press, Cambridge, MA.

[2] Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, 37(7):122-125.

[3] Bratman, M. E. (1987). *Intentions, Plans, and Practical Reason.* Harvard University Press: Cambridge, MA.

[4] Bratman, M. E., Israel, D. J., and Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:349-355.

[5] Cheikes, B. A. (1995). GIA: An Agent-Based Architecture for Intelligent Tutoring Systems. In *Proceedings of the CIKM '95 Workshop on Intelligent Information Agents*.

[6] Cohen, P. R. and Levesque, H. J. (1990). Intention is choice with commitment. *Artificial Intelligence*, 42:213-261.

[7] Dennett, D. C. (1987). *The Intentional Stance*. The MIT Press: Cambridge, MA.

[8] Emerson, E. A. (1990). Temporal and modal logic. In van Leeuwen, J., editor, *Handbook of Theoretical Computer Science*, pages 996-1072. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands.

[9] Finin, T., Weber, J,. Wiederhold, G., Genesereth, M., Fritzson, R., McKay, D., McGuire, J., Pelavin, R., Shapiro, S., and Beck, C. (1992). Specification of the KQML Agent-Communication Language. *Technical Report EIT TR 92-04*, Enterprise Integration Technologies, Palo Alto.

[10] Fitsilis P., Kameas A. and Pintelas P., (1996). ORIENTMAN: An intelligent tutor for the ORIENT software development methodology. *Software Engineering Journal*, July 1996, pp. 206-214.

[11] Genesereth, M. R., and Ketchpel, S. P. (1994). Software Agents. *Communications of the ACM*, 37(7).

[12] Georgeff, M. P. and Lansky, A. L. (1987). Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pp. 677-682, Seattle, WA.

[13] Gerogiannis V. C., Zaharakis I. D., Kameas A. D., Pintelas P. E. (1996). An Intelligent Tutoring System for the Yourdon Real-Time Systems Development Methodology, in *Proceedings of the First International Conference on Computers and Advanced Technologies in Education (CATE '96)*, March 18-20, 1996, Cairo, Egypt.

[14] Ikeda, M., and Mizoguchi, R., (1994). FITS: A Framework for ITS - a Computational Model of Tutoring. *Journal of Artificial Intelligence in Education*, 5 (3): 319-348.

[15] Jennings, N. R. (1993). Specification and implementation of a belief desire joint-intention architecture for collaborative problem solving. *Journal of Intelligent and Cooperative Information Systems*, 2(3): 289-318.

[16] Kameas A. D., and Pintelas P. E. (1996). The Functional Architecture and Interaction Model of a Generator of Intelligent Tutoring applications, *Journal on Systems and Software*, Elsevier Science Inc.

[17] McCarthy, J. (1978). Ascribing mental qualities to machines. *Technical report*, Stanford University AI Lab., Stanford, CA 94305.

[18] Minsky, M. (1994). A Conversation with Marvin Minsky About Agents. *Communications of the ACM*, 37(7).

[19] Minsky, M. (1985). *The Society of Mind.* Simon and Shuster, NY.

[20] Pollack, M. E. and Ringuette, M. (1990). Introducing the Tileworld: Experimentally evaluating agent architectures. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90),* pages 183-189, Boston, MA.

[21] Rao, A. S. and Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In Fikes, R. and Sandewall, E., editors, *Proceedings of Knowledge Representation and Reasoning (KR&R-91),* pp. 473-484. Morgan Kaufmann Publishers: San Mateo, CA.

[22] Rao A. S. and Georgeff M. P. (1991). Asymmetry thesis and side-effect problems in linear-time and branching-time intention logics, *Tech. Rep. 13,* Australian Artificial Intelligence Institute, Melbourne, Australia.

[23] Rao A. S. and Georgeff M. P. (1995). Formal models and decision procedures for multi-agent systems, *Tech. Rep. 61,* Australian Artificial Intelligence Institute, Melbourne, Australia.

[24] Rosenschein, S. and Kaelbling, L. P. (1986). The synthesis of digital machines with provable epistemic properties. In Halpern, J. Y., (ed.), *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge,* pp. 83-98. Morgan Kaufmann Publishers: San Mateo, CA.

[25] Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language.* Cambridge University Press: Cambridge, England.

[26] Seel, N. (1989). *Agent Theories and Architectures.* Ph.D. thesis, Surrey University, Guildford, UK.

[27] Shardlow, N. (1990). *Action and agency in cognitive science,* Master's thesis, Department of Psychology, University of Manchester, Oxford Rd, Manchester M13 9PL, UK.

[28] Shoham, Y. (1993). Agent-oriented programming. *Artificial Intelligence,* 60(1):51-92.

[29] Sonenberg E., Tidhar G., Werner E., Kinny D., Ljungberg M., and Rao A. (1992). Planned team activity, *Tech. Rep. 26,* Australian Artificial Intelligence Institute, Melbourne, Australia.

[30] Wooldridge, M. J., and Jennings N. R. (1995). Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review* 10 (2) 115-152.

[31] Wooldridge M. (1996). Practical Reasoning with Procedural Knowledge: A Logic of BDI Agents with Know - How. In *Proceedings of the International Conference on Formal and Applied Practical Reasoning.* Springer-Verlag.

[32] Zaharakis I. D., Kameas A. D. and Pintelas P. E., "MeT: The Expert Methodology Tutor of GENITOR". In *Microprocessing and Microprogramming,* 40, (10-12), 1994, pp. 855-860.

[33] Zaharakis I. D., Kameas A. D. and Pintelas P. E., "A Hybrid Expert System as an Embedded Module in Tutoring Systems", *TR 9502,* Division of Computational Mathematics and Informatics, Department of Mathematics, University of Patras, Hellas.