2016 IEEE 8th International Conference on Intelligent Systems

A Model of Extended BDI Agent with Autonomous Entities

(Integrating Autonomous Entities within BDI Agent)

Lada Maleš
Faculty of Humanities and Social Sciences
University of Split
21000 Split, Teslina 12, Croatia
lada.males@ffst.hr

Slobodan Ribarić
Faculty of Electrical Engineering and Computing
University of Zagreb
10000 Zagreb, Unska 3, Croatia
slobodan.ribaric@fer.hr

***Abstract*— In this paper a model of an extended BDI (EBDI) agent with autonomous entities and an architecture of the EBDI agent are introduced. The architecture consists of a beliefs module, a desires module, a decision generation module, an integration module, and a group of autonomous entities. In order to represent EBDI agent's mental attitudes (beliefs, desires, and intentions), we define L_m language of an agent's mental attitudes and describe a process of the intention creation. Furthermore, we define L_e language of an EBDI agent's mental attitudes content (spatial relations with temporal constraint and temporal relations). A disadvantage of the BDI model is the inability to adapt to changes in dynamic environment. When an unpredictable situation occurs and there is no applicable rule for that situation, an intention cannot be executed. Therefore, the EBDI model integrates agent with cognitively meaningful internal representation with computationally efficient autonomous entities. The purpose of the integration is to create an agent that possesses mental attitudes but it is adaptive in dynamic environments. When the EBDI agent does not have an applicable rule for current situation, the integration process supported by autonomous entities starts. Autonomous entities create a new representation of the dynamic environment (the new world model). The integration module takes the new world model, finds the solution and creates the new rule. The feasibility of our model has been validated with an example simulated in multiagent programmable modeling environment.**

Keywords—extended BDI agent; autonomous entities; hybrid agent;

I. INTRODUCTION

Logical foundations of agency have been established since late 1980s, but nevertheless multiagent systems models built on logic are still popular [1], [2]. The reason is that logic can be a powerful tool for reasoning about multiagent systems [3].Van der Hoek and Wooldridge present the following reasons for that [3]: (i) logic provides a language for specification of the properties of the agents, as well as properties of the environment; (ii) properties expressed in a form of logical formulas are part of the inference system, so they can be used to deduce other properties; (iii) logic provides a formal semantics in which the sentences from the language are assigned a precise meaning.

One of the most popular agent models is the Belief-Desire-Intention (BDI) model [4], [5], [6], which comes from cognitive science and philosophy.

The BDI agent has mental attitudes: beliefs, desires, and intentions. Beliefs represent the informational state of the BDI agent i.e. what it knows about itself and the world. Desires (or goals) are its motivational state, that is, what the agent prefers. Intentions represent the deliberative state of the agent and they tend to lead to action. BDI agents have been used to model human behavior and create human-like characters in simulated environments [7], [8], [9], [10], [11]. One of the main limitations of the BDI agent is that it does not possess the ability to adapt to a new situation.

An adaptation is important in dynamic environment, since changes in environment can cause the BDI agent's plans to become inefficient or inapplicable. Since multiagent systems based on BDI agents cannot be used for modelling situations in dynamic environments, they are often integrated with various methods and form a hybrid agent model.

Therefore, various hybrid agent models have been introduced. A hybrid agent's architecture model, implementing CBR (Case Based Reasoning) and CBP (Case Based Planning) reasoning mechanisms into deliberative BDI agents is presented in [12], [13]. The temporal bounded agent version of the CBP-BDI agent is the TB-CBP-BDI agent that is a real-time intelligent agent providing special abilities for planning in a predictable time [14], [15]. Another interesting approach is combining BDI and MDP (Markov Decision Processes) [16]. Hybrid BDI-PGM framework combines PGM (Probabilistic Graphical Model) and BDI [17]. An example of hybrid architecture that integrates the BDI model with low-level reinforcement learning algorithms is BDI-FALCON [18]. It extends a neural network based reinforcement learner into a deliberative reasoner. In [19] a number of different approaches for integrating swarming and BDI agents are described. There are two extreme types of agent: "heavyweight" and "lightweight" agents. The first type of agents originates from artificial intelligence and cognitive science. Internal representation is symbolic and manipulated with some form of logic. The strength of this type of agent is that they are inspired by human concept of knowledge and deliberation, they use cogni-

978-1-5090-1353-1/16/$31.00 ©2016

tive constructs that are meaningful to people and that make them intelligible for humans. A disadvantage of this type of agents is that the process of logical computation is often intractable, undecidable, and brittle. The second type of agents originates from the study of animal behavior. Since 1990s, swarm intelligence has become increasingly popular [20] [21]. The agents emulate animal behavior where emphasis is not on individual behavior but on society as a whole that might be described as intelligent. An internal representation is numerical so they use optimization methods for exploring parameter space. This approach makes them computationally efficient on one side, and on the other, they are difficult to understand from human stand.

An integration of different type of agents in a single system might be a challenge since they possess different cognitive levels. In our approach we integrate “lightweight” agents (we have called them autonomous entities) within a “heavyweight” agent (we have called it EBDI agent) to solve specific problems.

The rest of this paper is organized as follows. In Section 2, the architecture of an extended BDI agent is introduced. A modal logic representation of the extended BDI agent is described in Section 3. In Section 4, the way in which autonomous entities are integrated in BDI architecture is introduced. An example of our model is given in Section 5, with simulation in a multiagent programmable environment (please refer to http://www.ffst.unist.hr/~lmales/IS16/).

II. ARCHITECURE OF EXTENDED BDI AGENT

The architecture of an extended BDI agent consists of a beliefs module, desires module, a decisions generation module, an integration module, and autonomous entities (shown in Fig. 1). In further text, extended BDI agent will be referred to as EBDI agent.

The beliefs module creates beliefs about the world through perception. It takes perception from sensors and processes them. The data obtained by perception is processed according to an agent’s needs, meaning that irrelevant data is discarded. The results of the perception processing are beliefs. A perception of an agent can be realistic when an agent has perfect sensors. However, when sensors are imperfect, they give a modified “picture” of the world. For example, a person who does not see or hear well has a picture of the world that is not realistic. The agent’s beliefs have influence on the intention creation process (deliberation) in the decision generation module.

The desires module generates desires of an agent. It consists of a desires generator and a desire. A desire should be consistent with beliefs. If they are not consistent, a desire is discarded by decision generation module and sent back for revision in desires module. Like beliefs, desires also have influence on the intention creation process.

The decision generation module creates an intention that leads to action. An agent’s deliberation is a process of intention creation. The agent deliberates upon current beliefs and desires, as well as upon adequate rule from the base of rules that matches current beliefs and desires. A base of rules contains rules where antecedent is a conjunction of beliefs and desires, and the consequent is an intention. If an applicable rule exists, deliberation creates intention; otherwise, an integration module is actuated. An applicable rule is the rule that an agent can apply, i.e. spatio-temporal and temporal constraints in relations are satisfied. An intention is a part of plan that should be executed immediately or with time delay according to temporal constraint of the intention. Since the EBDI agent acts in dynamic environment, there is a possibility that unpredictable changes in the system might occur. If there is no rule that matches current beliefs and desires, e.g. the agent cannot create an intention. In this case, the agent needs to revise the desire or to create new rules for the agent. For this purpose, the decision generation module induces an integration module.

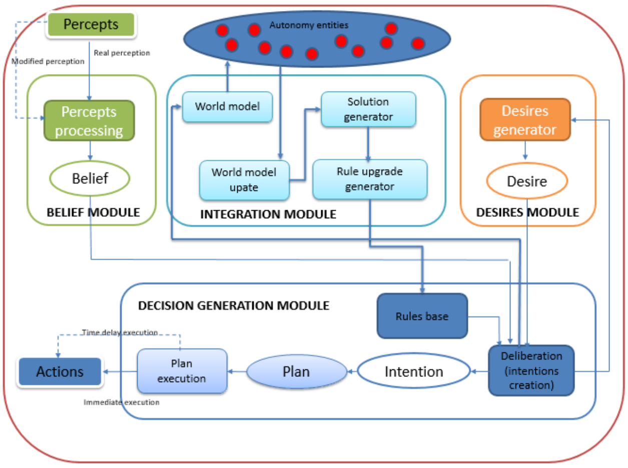

Figure 1. The architecture of the EBDI agent

An extension of the BDI agent is the integration module that uses services of autonomous entities. The integration module consists of the following elements: a world model, a world model update, a solution generator, and a rule-upgrade generator. Each time when an intention cannot be created, an update of the world model needs to be done. Current world model (map of the environment that depends on an agent’s current beliefs about the environment) is given to a group of autonomous entities. They act in the dynamic environment, where they perform primitive behaviors according to their behavior rules. Autonomous entities can move within the environment, put traces at elements of the environment, read traces put by other autonomous entities, and behave accordingly. A result of their actions is the updated world model (a new map of the environment). With this information, a solution generator finds a solution (e.g. the shortest path with some of the well-known algorithms). After that, the rules need to be upgraded and delivered to the rules base.

If the environment can change while an agent is deliberating, the environment is dynamic; otherwise, it is static. In static environments an agent need not keep looking at the world and worry about the passage of time, while it is deciding on an action [22]. In order to EBDI agent properly function in a dynamic environment, it needs to be aware of other agents and objects, relations between them, as well as temporal constraints of these relations.

The EBDI agent is an active participant of an environment, while an object is a passive element of an environment. Therefore, an agent autonomously changes its relations with other agents, unlike an object. An EBDI agent could be in many different types of relations with other EBDI agents and objects (i.e. the tourist is in front of the museum during a period of 10 minutes or the tourist gets off the bus at nine o'clock), depending on the problem domain. In the paper, we have introduced spatial relations with temporal constraints and combined them with temporal relations [23] to model situations in the dynamic environment. Spatial relations with temporal constraint are easy to understand because they are part of everyday human life. Other types of relations with temporal constraint could be integrated in similar way.

III. MODAL LOGIC REPRESENTATION OF EBDI AGENT

The EBDI agent presented in this paper has mental attitudes (beliefs, desires and intentions) and extension that enables him to use services of autonomous entities. Beliefs, desires and intentions are products of beliefs, desires and decision generator modules. An integration module uses services of autonomous entities and gives the EBDI agent ability to react on changes in the environment.

The EBDI agent's mental attitudes are denoted by modal logic operators: *BEL*, *DES* and *INT*. The content of agent's mental attitudes are spatial relation with temporal constraint and temporal relation. In the rest of the paper, spatial relation with temporal constraint and temporal relation will be denoted with abbreviations *srtc* and *tr*.

A *srtc* describes spatial relation between two EBDI agents, and among the EBDI agent and the object that exists during temporal interval or time point. The relation is written in a form of 3-place predicate, where arguments are agents, object and time. A *tr* describes the relation between two temporal intervals, a temporal interval and a time point and two time points. The relation is written in a form on 2-place predicate, where arguments are temporal intervals or time points. EBDI agents are temporally omniscient, they have true beliefs about temporal relations.

In other words, we could say that an EBDI agent with mental attitudes believes in *srtc* and *tr* relations, desire and has intention to achieve *srtc* relations.

In order to represent agent's mental attitudes we define language syntax L_m, but first language syntax of mental attitude content L_e must be introduced. The language L_e is used to represent a spatial knowledge with temporal constraint and temporal knowledge.

A. L_e language of an agent's mental attitudes content

First, a vocabulary of the L_e is defined. Terms are (i) variables $v_1,\ldots,v_n$: agents a_1, a_2, …; temporal intervals τ_1, τ_2, …, time points t_1, t_2, ...; and objects in environment o_1, o_2, ... (ii) constants (which are introduced as language extension when needed) (iii) predicates. Logical constants are standardly defined. Atomic formula is defined as predicate $P(v_1, \ldots, v_n)$. We suggest two types of atomic formulas in L_e:

- for n=2 an atomic formula is called temporal relation atomic formula and arguments are temporal variables (temporal intervals or time points)
- for n=3 an atomic formula is called spatial relation with temporal constraint atomic formula and arguments are agents, object in environment and temporal variables (temporal intervals or time points).

A mental attitude content language formula φ is defined in (1)

$$\varphi ::= P(v_1, \ldots v_n) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \forall v\varphi \qquad (1)$$

where predicate $P(v_1, \ldots v_n)$ is *temporal relation atomic formula* or *spatial relation with temporal constraint atomic formula.*

Temporal atomic formulas and spatial relation with temporal constraint atomic formulas are given below.

Definition 1. *Primitive temporal relation formulas (predicates)*

Let τ be a temporal interval and t time point. Relations between two temporal interval, interval and time point and two time points are defined by temporal relation formulas (predicates). Primitive temporal relation predicates are: Meets(τ_1, τ_2), Before(τ_1, τ_2), During(τ_1, τ_2), Overlap(τ_1, τ_2), Starts(τ1, τ2), Finishes(τ_1, τ_2) and Equal(τ_1, τ_2) [23]; Meets(t, τ), Before(t, τ), During(t, τ), Starts(t, τ) and Finishes(t, τ); Before(t_1, t_2) and Equal(t_1, t_2)

Definition 2. T*emporal relation formulas (predicates)*

Other temporal relation predicates are defined by primitive temporal predicates: MetBy(τ_1, τ_2) iff Meets(τ_2, τ_1), After(τ_1, τ_2) iff Before(τ_2, τ_1), OverlapBy(τ_1, τ_2) iff Overlap(τ_2, τ_1), StartsBy(τ_1, τ_2) iff Start(τ_2, τ_1), FinishBy(τ_1, τ_2) iff Finishes(τ_2, τ_1); MetBy(τ, t) iff Meets(t, τ), After(τ, t) iff Before(t, τ); After(t_1, t_2) iff Before(t_2, t_1)

Axiom 1. *Spatial order with temporal constraint*

Let a_i, a_j be EBDI agents and o objects. Then $\forall a_i, a_j, o$ ($a_i \neq a_j$) at least one spatial relation with temporal constraint On, LeftOf or FrontOf holds i.e. each agent must be in at least one relation with other agent or object. These three relations are primitive and primitive spatial relation with temporal constraint predicates describes them. Other *srtc* predicates are derived from them.

Definition 3. *Primitive spatial relation with temporal constraint formulas (predicates)*

Let a_i, a_j be EBDI agents, o objects and τ temporal constraint (temporal interval or time point), where $\tau=[t_1, \ldots, t_2]$, $t_1 \leq t_2$. Primitive spatial relation with temporal constraint predicates are On(a_i, a_j, τ), On(a, o, τ), LeftOf(a_i, a_j, τ), LeftOf(a, o, τ), FrontOf(a_i, a_j, τ) and FrontOf(a, o, τ).

For primitive predicate On(a, o, τ) the following properties hold: seriality $\forall a_i \exists o$, holds On(a_i, o, τ) - agent must be in relation with some object in environment and transitivity $\forall$ a_i, a_j and o, $a_i \neq a_j$, holds $\text{On}(a_i, a_j, \tau) \wedge \text{On}(a_j, o, \tau) \rightarrow \text{On}(a_i, o, \tau)$.

A primitive predicate $On(a, o, \tau)$ meaning that an agent a is on, at or in object o, i.e. an interpretation of primitive predicate depends of prepositions used in English grammar. e.g. tourist is during period τ at the station, on the sidewalk, at the museum etc. A primitive predicate $On(a_i, a_j, \tau)$ meaning that agent a_i is on or in agent a_j during period τ. Consequently, they share same space (element of environment) during time period τ, e.g. tourist is during period τ in the car, on the bus etc.

For primitive predicate $On(a_i, a_j, \tau)$ the following properties hold: transitivity $\forall$ a_i, a_j, a_k, $a_i \neq a_j \neq a_k$, holds $On(a_i, a_j, \tau) \wedge On(a_j, a_k, \tau) \rightarrow On(a_i, a_k, \tau)$, Euclidian property (a weakened form of transitivity) $\forall$ a_i, a_j, a_k, $a_i \neq a_j \neq a_k$, holds $On(a_i, a_j, \tau) \wedge On(a_i, a_k, \tau) \rightarrow On(a_k, a_j, \tau)$ or $On(a_i, a_j, \tau) \wedge On(a_i, a_k, \tau) \rightarrow On(a_j, a_k, \tau)$ and irreflexivity $\forall a_i$, holds $\neg On(a_i, a_i, \tau)$ - agent can't be in relation with himself.

For primitive predicates $LeftOf(a_i, a_j, \tau)$ and $LeftOf(a, o, \tau)$ the following properties hold: transitivity $\forall$ a_i, a_j, a_k and o, $a_i \neq a_j \neq a_k$, holds $LeftOf(a_i, a_j, \tau) \wedge LeftOf(a_j, a_k, \tau) \rightarrow LeftOf(a_i, a_k, \tau)$ and $LeftOf(a_i, o, \tau) \wedge LeftOf(o, a_j, \tau) \rightarrow LeftOf(a_i, a_j, \tau)$ and irreflexivity $\forall a_i$, holds $\neg On(a_i, a_i, \tau)$.

A primitive predicate $LeftOf(a_i, a_j, \tau)$ meaning that an agent a_i is left of an agent a_j, while $LeftOf(a, o, \tau)$ meaning that an agent a is left of an object o.

Primitive predicates FrontOf(ai, aj, τ) and $FrontOf(a, o, \tau)$ the following properties hold: transitivity $\forall$ a_i, a_j, a_k and o, $a_i \neq a_j \neq a_k$, holds $FrontOf(a_i, a_j, \tau) \wedge FrontOf(a_j, a_k, \tau) \rightarrow FrontOf(a_i, a_k, \tau)$ and $FrontOf(a_i, o, \tau) \wedge FrontOf(o, a_j, \tau) \rightarrow FrontOf(a_i, a_j, \tau)$ and irreflexivity $\forall a_i$, holds $\neg$ $FrontOf(a_i, a_i, \tau)$.

A primitive predicate $FrontOf(a_i, a_j, \tau)$ meaning that an agent a_i is in front of an agent a_j, while $FrontOf(a, o, \tau)$ meaning that an agent a is in front of an object o.

Definition 4. *Spatial relation with temporal constraint predicates*

Other *srtc* predicates are defined by primitive predicates: $RightOf(a_i, a_j, \tau)$ iff $LeftOf(a_j, a_i, \tau)$. $Behind(a_i, a_j, \tau)$ iff $FrontOf(a_j, a_i, \tau)$, $LeftFrontOf(a_i, a_j, \tau)$ iff $LeftOf(a_j, a_i, \tau) \wedge FrontOf(a_i, a_j, \tau)$, etc.

Definition 5. *Dynamic srtc formulas (predicates)*

A spatial relation with temporal constrain *srtc* could be static or dynamic. In static *srtc* agent does not changes its position in space during time. All predicates introduced above define static relations. Dynamic relations describe agent's movement and they are defined by dynamic predicates. Moreover, one of the agent's dynamic properties is a possibility to make transition i.e. to disappear from the system and to appear again in the system. When an EBDI agent disappears from system, the EBDI agent inherits *srtc* of another EBDI agent or object in which it appears. Conversely, when the EBDI agent reappears in the system, *srtc* of that another EBDI agent or object in which it was part of is no longer valid.

Let a_1 be agent, o_1 and o_2 object and t_1 and t_2 time points. Dynamic predicates $Forward(a, o, t)$, $Backward(a, o, t)$, $TurnLeft(a, o, t)$ and $TurnRight(a, o, t)$ are defined as follows:

- $Forward(a_1, o_1, t_1)$ iff $On(a_1, o_1, t_1) \wedge FrontOf(a_1, o_2, t_1) \wedge On(a_1, o_2, t_2) \wedge Meet(t_1, t_2)$
- $Backward(a_1, o_1, t_1)$ iff $On(a_1, o_1, t_1) \wedge BackOf(a_1, o_2, t_1) \wedge On(a_1, o_2, t_2) \wedge Meet(t_1, t_2)$
- $TurnLeft(a_1, o_1, t_1)$ iff $On(a_1, o_1, t_1) \wedge LeftOf(a_1, o_2, t_1) \wedge FrontOf(a_1, o_2, t_2) \wedge Meet(t_1, t_2)$
- $TurnRight(a_1, o_1, t_1)$ iff $On(a_1, o_1, t_1) \wedge RightOf(a_1, o_2, t_1) \wedge FrontOf(a_1, o_2, t_2) \wedge Meet(t_1, t_2)$

Let a_1 and a_2 be agents, o object, τ temporal interval and t_1 and t_2 time points. Dynamic predicates $Closer(a_1, a_2, \tau)$, $Closer(a, o, \tau)$ and $Further(a_1, a_2, t)$ and $Further(a, o, \tau)$ are defined as follows:

- $Closer(a_1, o, \tau)$ iff $FrontOf(a_1, o, t_1) \wedge Forward(a_1, o, t_1) \wedge Starts(t_1, \tau) \wedge Finishes(t_2, \tau) \wedge FrontOf(a_1, o, t_2)$
- $Closer(a_1, a_2, \tau)$ iff $FrontOf(a_1, a_2, t_1) \wedge Forward(a_1, a_2, t_1) \wedge Starts(t_1, \tau) \wedge Finishes(t_2, \tau) \wedge FrontOf(a_1, a_2, t_2)$
- $Further(a_1, o, \tau)$ iff $Behind(a_1, o, t_1) \wedge Backward(a_1, o, t_1) \wedge Starts(t_1, \tau) \wedge Finishes(t_2, \tau) \wedge Behind(a_1, o, t_2)$
- $Further(a_1, a_2, \tau)$ iff $Behind(a_1, a_2, t_1) \wedge Backward(a_1, a_2, t_1) \wedge Starts(t_1, \tau) \wedge Finishes(t_2, \tau) \wedge Behind(a_1, a_2, t_2)$

Let a_1 and a_2 be agents, o object and t_1 and t_2 time points. Dynamic predicates that describe transition are $Disappear\ (a_1, a_2, t)$, $Disappear\ (a, o, t)$, $Appear\ (a_1, a_2, t)$ and $Appear\ (a_1, a_2, t)$:

- $Disappear\ (a_1, a_2, t_2)$ iff $On\ (a_1, o, t_1) \wedge On(a_2, o, t_1) \wedge On(a_2, o, t_2) \wedge Meet(t_1, t_2)$
- $Disappear\ (a_1, o, t_2)$ iff $On(a_1, o, t_1) \wedge Meet(t_1, t_2)$
- $Appear(a_1, a_2, t_2)$ iff $On(a_2, o, t_1) \wedge On(a_1, o, t_2) \wedge On(a_2, o, t_2) \wedge Meet(t_1, t_2)$
- $Appear(a_1, o, t_2)$ iff $On(a_1, o, t_1) \wedge Meet(t_1, t_2)$

Remark: Instead of primitive predicates On, predicates FrontOf and LeftOf can be used.

B. L_m language of an agent's mental attitudes

L_e language enables description of spatial relations with temporal constraint and temporal relation. For description of mental attitudes of an EBDI agent, L_m language has been used [24]. The language L_m is built over L_e by use of modal operators: *BEL*, *DES* and *INT*.

Let φ be a formula of L_e and φ' is formula φ with n=3, defined in (1). A formula ψ of L_m is defined in (2):

$$\psi := BEL\ \varphi \mid DES\ \varphi' \mid INT\ \varphi' \mid \neg\psi \mid (\psi_1 \wedge \psi_2) \mid \forall v \psi. \quad (2)$$

L_a is a restricted language of quantified modal logic in which iteration of modal operators is not allowed e.g. $BEL(DES\,\varphi)$.

An EBDI agent's mental attitude has temporal constraint. Temporal duration of agent's beliefs, desires and intentions are written in formula φ'.

C. Beliefs, desires and intentions

There are three types of the EBDI agent beliefs: domain oriented beliefs, fact oriented beliefs and inferential beliefs. First, we need to distinguish knowledge and belief. EBDI agent's beliefs are not necessary the truth. The EBDI agent presented in this paper believes in *srtc* and *tr* relations, but he also must have beliefs about a domain of the problem. The EBDI agent's believes about other EBDI agents and objects (for example, their appearance or position in environment) are included in domain-oriented beliefs. For creating other two types of beliefs, these beliefs are necessary. For instance, the EBDI agent believes that a taxi is yellow, a public bus is blue etc. because without those beliefs he could not recognize them. Also EBDI agent believes in the temporal order e.g. the time instant of buying tickets is before the temporal interval when it is at the cinema. The focus of this paper are *spatial relations with temporal constraint* so EBDI agent's domain oriented beliefs are assumed to be supported in a multiagent system simulator. Fact oriented beliefs are acquired by perception. Inferential beliefs are combination of two previously mentioned beliefs.

Example 1. The EBDI agent a_1 believes that it will be in front of an object o_1 at time point t during temporal interval τ. Values Mary and museum are assigned to the argument a_1 and o_1, while the values of t and τ will be assigned to the arguments during an execution of the program, because they are part of EBDI agent's deliberation. The formula is $BEL(\text{FrontOf}(\text{Mary}, \text{museum}, t) \wedge \text{During}(t, \tau))$.

Axiom 1. The modal operator *BEL* satisfies these properties: necessity rule and axioms K, D and T. The necessity rule $\varphi \rightarrow BEL\ \varphi$ said that if something is true than the EBDI agent believes that is the true. The axiom K (sometimes called modal modus ponens) $(BEL\ \varphi_1 \wedge BEL\ (\varphi_1 \rightarrow \varphi_2)) \rightarrow BEL\ \varphi_2$, means that the EBDI agent believes in the consequences of his believes. The axiom D $\neg BEL(\varphi \wedge \neg\varphi)$ means that the EBDI agent doesn't believe in contradiction. The axiom T $BEL\ \varphi \rightarrow \varphi$ interpretation states that what the EBDI agent believes is the case. The axiom T is valid for fact-oriented beliefs and domain oriented beliefs acquired by perception and interaction.

Desires are the EBDI agent's motivations and its goals. The EBDI agent wants to accomplish the *srtc* with other agents or objects. The EBDI agent has no desire of a *tr*.

Example 2. An EBDI agent a_1 wants to be on/at/in object o during the interval τ. The values Mary, museum and [10:00, 11:00] are assigned to the arguments a_1, o_1 and τ: $DES\ (\text{On}(\text{Mary}, \text{museum}, [10{:}00, 11{:}00]))$.

Axiom 2. The modal operator *DES* satisfies properties: necessity rule and axioms K and D. The necessity rule $\varphi \rightarrow DES\ \varphi$ said that if something is truth than EBDI agent desires that is the truth. The axiom K $(DES\ \varphi_1 \wedge DES\ (\varphi_1 \rightarrow \varphi_2)) \rightarrow DES\ \varphi_2$, means that agent desires the consequences of its desires. The axiom D $\neg DES(\varphi \wedge \neg\varphi)$ means that the EBDI agent could not desire a contradiction.

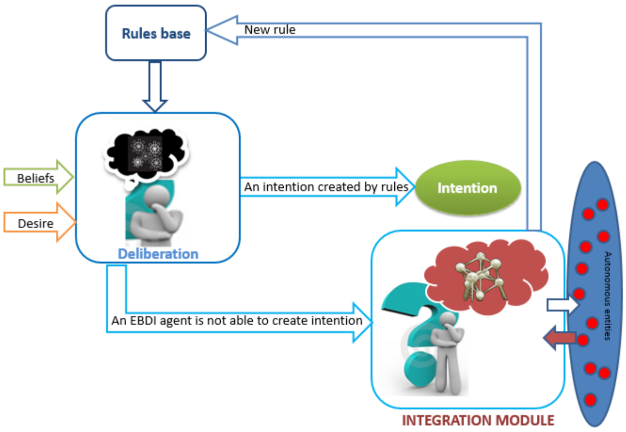

Figure 2. The integration process

The EBDI agent has an intention to accomplish the *srtc* with the other EBDI agent and object. The EBDI agent has no intention of achieving *tr*, because they describe a temporal relationship between agents in the srtc relation

Example 3. The EBDI agent a_1 has an intention to enter on/in/at the EBDI agent a_2 at time point t_1 and to go outside of an agent a_2 at time point t_2. The values assigned to the variables a_1 and a_2 are John and bus_no_5: $INT(\text{Disappear}(\text{John}, \text{bus\_no\_5}, t_1) \wedge INT(\text{Appear}(\text{John}, \text{bus\_no\_5}, t_2))$. The values for the variables t_1 and t_2 depend of the behavior of other EBDI agents and EBDI agent's intention creation process, therefore they are assigned dynamically i.e. when the bus_no_5 is at the bus station and John believes that. Moreover, during the temporal interval $\tau=[t_1, \ldots, t_2]$, John is on the bus_no_5, but he disappears from system and he is not autonomous. Otherwise, John could be autonomous and the formula is then $INT(\text{On}(\text{John}, \text{bus\_no\_5}, \tau)$.

Axiom 3. The modal operator *INT* satisfies properties these properties: necessity rule and axioms K and D. The necessity rule $\varphi \rightarrow INT\ \varphi$ said that if something is the truth than agent has an intention for the truth. The axiom K $(INT\ \varphi_1 \wedge INT\ (\varphi_1 \rightarrow \varphi_2)) \rightarrow INT\ \varphi_2$, means that EBDI agent has an intention to do all consequences of its intentions. The axiom D $\neg INT(\varphi \wedge \neg\varphi)$ meaning that agent couldn't has an intention that is a contradiction.

D. A process of an intention creation

The EBDI can deliberate about its situation when it possesses domain-oriented beliefs, when it believes the perception of its current situation and has its own desires. Then it creates an intention that immediately leads to action or with time delay. In same situation, different agents with equal beliefs can create a different intention since their desires are different.

Intention creation rules depict situations that we suppose the EBDI agent can be part of. In that way the EBDI agent's behavior is determined in predictable situations. A rule for creating the intention is a quantified conditional, where antecedent is a conjunction of modals *BEL* and *DES* and a consequent is *INT* modal. A content of modal operator *BEL* is a conjunction of *srtc* and *tr*. A content of modal operators *DES* and *INT* is *srtc*.

The form of the rule for creating the intention is defined in (3):

$$BEL(srtc_1 \wedge srtc_2 \wedge \ldots\ srtc_n \wedge tr_1 \wedge \ldots \wedge tr_m) \wedge DES(srtc_p) \rightarrow INT(srtc_q) \quad (3)$$

for m≥1, n≥1, where *srtc* can be any spatial relation with temporal constraint and *tr* can be any temporal relation.

IV. INTEGRATING AUTONOMOUS ENTITIES WITHIN EBDI AGENT

In the previous section, it has been explained the way an EBDI agent creates an intention. However, this is not enough for EBDI agent to adapt to changes in environment. In this section, an EBDI agent's integration process is presented. The process is carried out in the integration module and supported by autonomous entities (Fig. 1). The goal is to create new rules for creation of intentions and make an EBDI agent adaptable to changes in the environment.

An interaction of a decision generation module and an integration module is illustrated in (Fig. 2). The process of integration starts when an EBDI agent cannot create intention from current believes, desires and rules. A decision generation module induces an integration module and the result is new rule/rules in the rules base. An integration module takes a current world model and asks autonomous entities for services. With updated world model (the new map of the environment) created by autonomous entities, the integration module finds a solution, transforms it into new rules and delivers them to the rules base in the decision generation module. Autonomous entities create a new world model. In order to accomplish this task they possess some features of autonomy oriented entities [25], such as change of environment state and death of entities. Furthermore, the entities use procedures inspired by nature e.g. put traces in environment. Autonomous entities behave accordance to information left within the environment.

A. Model of an autonomous entity

An integration module gives the current world model i.e. the map of an environment to autonomy entities $e_i \in E$ (where E is a set of autonomous entities). Autonomous entities move from one element of the map of the environment to another, turn to one of four possible directions, put traces on elements of the map and read traces from elements of the map. After some time autonomous entities die i.e. have a time of life. An interaction between autonomous entities is indirect, through traces on the element of the map of the environment.

The map of the environment is a two-dimensional square grid. A set D is a set of the elements $d_i \in D$ of the environment. Every $d_i \in D$ is a represented as a square on the map of the environment and described with (i) x and y coordinate in the grid (position of the center of the square within the map of the environment) (ii) traces that autonomous entities put on the element of the environment d_i. There are two types of traces: (i) trace_o described by quadruplets (s_1, s_2, s_3, s_4) whose elements are probability that the autonomous entity will turn to the one of four possible directions (ii) trace_m marks the element of the environment.

The autonomous entity is a pair (C, F) where C describes a state of the autonomous entity e_i and F describes autonomous entity's primitive behaviors. State C of the autonomous entity is characterized by two attributes: a current position of the autonomous entity on the element of the environment $d_i \in D$ (x and y coordinate in the grid) and the time of life (natural number >0). On every step of the autonomous entity within the map of the environment, the time of life is decreased by one. When the number is equal to 0, the autonomous entity dies.

A set F=*{fto,ftm}* describes a primitive behavior of autonomous entities. Function *fto* enables autonomous entities to turn randomly to one of the four possible direction on the map of the environment, therefore moving of autonomy entities is not deterministic. Function *ftm* denotes on the map of the environment where are the obstacles and where are not.

Function *fto*, maps element of environment $d_i \in D$ into a set S of quadruplets (s_1, s_2, s_3, s_4) (4). Elements of quadruplets are real number from open interval (0, 1) and $\Sigma_i s_i = 1$, i=1..4.

$$fto\colon D \rightarrow S \quad (4)$$

A value of every element s_i in quadruplets (s_1, s_2, s_3, s_4) is a possibility for autonomous entities to turn to one of the four directions (i.e. s_1 on the left, s_2 on the right, s_3 forward (keep the same direction) and s_4 backward). An initial value is 0.25 for each s_i. When the autonomy entity is on the element d_i, it reads values from the quadruplets and turns. If there is an obstacle in front of it calculates a new values of quadruplets i.e. (s'_1, s'_2, s'_3, s'_4) and turn again, calculates again etc. until it turns in the direction without obstacle, then it moves further. A value s'_i is the new value for each of four directions, where k is a current possibility for the direction with obstacle divided with number 4 since d_i has four neighbors (5).

$$s'_i = \left\{ \begin{matrix} k\text{ ; for the direction with the obstacle, where } k = \frac{s_i}{4} \\ s_i + k\,\frac{3}{4}\text{ ; for other three directions} \end{matrix} \right\} \quad (5)$$

Every time when the autonomous entity is in front of an obstacle the possibility for that direction decrease, and for other three directions increase. The purpose of the quadruplets on every element d_i is to increase possibility for next autonomous entities to turn into direction without an obstacle. Consequently, autonomy entities move faster within unknown environment.

Function *ftm*, *ftm*: D → M, maps an element of the environment into a set M, where M is the set of marks e.g. M={x,

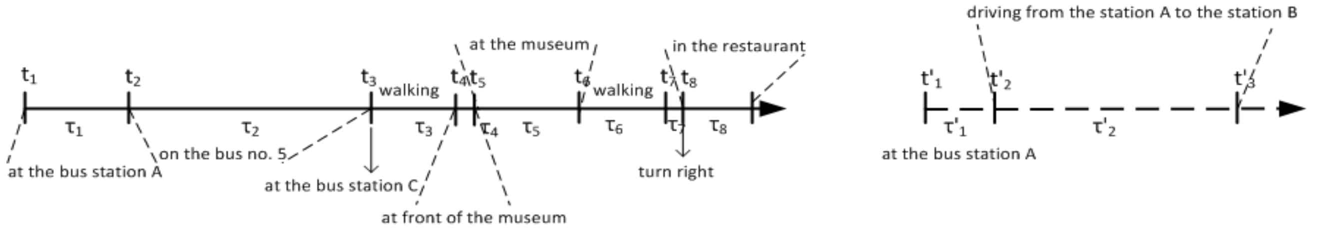

Figure 3. Temporal relations of the tourist and the bus no. 5 from the example

0, 1, S, G}. For example, a mark x is an initial mark, assigned to every element of the environment at the beginning of the marking process. A mark 0 describes an element of the environment with no obstacle and mark 1 describes an element of the environment with an obstacle. Marks S and G symbolize the start and goal position. A marking process starts with world model presented by integration module. Autonomous entities behave in the following way, when they are in front of the obstacle, they marked that element with 1, when the element in front of them has no obstacle they mark them with 0. After that they turned in random direction according to function *fto* and proceed marking.

When the new world model (the new map of the environment) is finished, autonomous entities deliver it to integration module, where the solution generator finds a shortest path with well-known algorithms. In this case, we used wavefront algorithm [26]. After that, new rules could be written according to the solution and delivered to the rules base in decision generation module.

V. EXAMPLE

For validation of our model, we have chosen an example that is simple and familiar. Nevertheless, even for this simple example numerous *srtc* and *tr* could be identified.

A. *A problem modelling*

A tourist wants to visit a museum and after that, wants to have lunch in the restaurant, which is at the walking distance from the museum. In order to do this he needs to take a bus number 5 at the station A, get off the bus at the station C, take a short walk to the museum and enter at the museum. After visiting the museum, the tourist goes to the restaurant. All the time the tourist must be aware of temporal constraint e.g. tourist would not wait a bus if he/she believes that there is not enough time to arrive at the museum on time. In that case he would rather goes to the restaurant first and then to the museum. The tourist knows what is critical time t*is to accomplish the original plan.

Relevant *srtc* relations for the tourist in this example are: On (tourist, station A, t_1) – the tourist arrives at the station A at time point t_1; On(tourist, station A, τ_1) – the tourist is at the station A during a period τ_1; Disappear(tourist, bus no 5, t_2) – the tourist gets on the bus no 5 at time point t_2; Appear(tourist, bus no 5, t_3) – the tourist get off the bus at time point t_3; On(tourist, station B, t_3) – the tourist is at the station B at time point t_3; Closer(tourist, museum, τ_3) – the tourist takes a walk to the museum during period τ_3; FrontOf(tourist, museum, τ_4) – the tourist is at the front of the museum during period τ_4; On(tourist, museum, τ_5) – the tourist is at the museum during time period τ_5; FrontOf(tourist, museum, t_6) – the tourist is in front of the museum at time point t_6; Closer (tourist, restaurant, τ_6) – the tourist takes a walk to the restaurant; LeftOf(tourist, restaurant, τ_7) – the tourist is on the left side of the restaurant during period τ_7; TurnRight(tourist, restaurant, t_8) – the tourist turn right in front of the restaurant at the time point t_8 and On(tourist, restaurant, τ_8) – the tourist is at the restaurant during period τ_8.

Relevant *srtc* relations for the bus in this example are: On(bus no 5, station A, t'_1) – the bus no 5 arrives at the station A at time point t'_1; On(bus no 5, station A, τ'_1) – the bus no 5 is at the station during a period τ'_1; Forward(bus no 5, road, τ'_2) – the bus no 5 is on the road during period τ'_2 and On(bus no 5, station B, t'_3) – the bus no 5 arrives at the station B at time point t'_3

Temporal relations for the tourist and the bus are shown in Fig. 3.

B. *Rules for creating intentions*

In further text, rules for creating intentions are presented.

Rule 1. The tourist is at the station A and the bus no. 5 is not at the station A, so the tourist creates an intention to wait for the bus. Also a period of waiting must precede critical time t*. The rule 1 is *BEL*(On (tourist, station A, t_1) $\wedge$ $\neg$On (bus no 5, station A, t_1) $\wedge$ Starts(t_1, τ_1) $\wedge$ Before(τ_1, t*) $\wedge$ Before(t*, τ_5)) $\wedge$ *DES*(On(tourist, museum, τ_5)) $\rightarrow$ *INT*(On(tourist, station A, τ_1))

Rule 2a. The bus no 5 arrives at the station A, the tourist is at the station waiting for the bus and creates intention to get on the bus. The rule 2a is *BEL*(On (tourist, station A, τ_1) $\wedge$ On (bus no 5, station A, t'_1) $\wedge$ During(t'_1, τ_1) $\wedge$ During(t_2, τ'_1) $\wedge$ Finishes(t_2, τ_1) $\wedge$ Before(t_2, t*) $\wedge$ Before (t*, τ_5)) $\wedge$ *DES*(On(tourist, museum, τ_5)) $\rightarrow$ *INT*(Disappear(tourist, bus no 5, t_2))

Rule 2b. The bus no 5 is already at the station A when the tourist arrives at the station. The tourist gets on the bus. The rule 2b is *BEL*(On (tourist, station A, t_1) $\wedge$ On (bus no 5, station A, τ'_1) $\wedge$ During(t_1, τ'_1) $\wedge$ Before(t_1, t*) $\wedge$ Before(t_*, τ_5)) $\wedge$ *DES*(On(tourist, museum, τ_5)) $\rightarrow$ *INT*(Disappear(tourist, bus no 5, t_2)); (for this situation t_1=t_2)

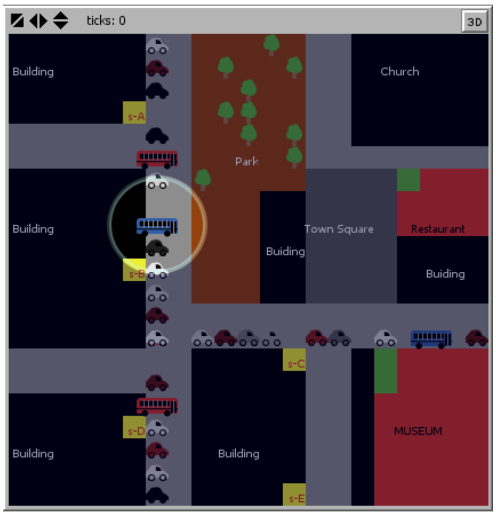

Figure 4. Screenshot of Netlogo simulation (the tourist is on the bus no 5 marked by circle)

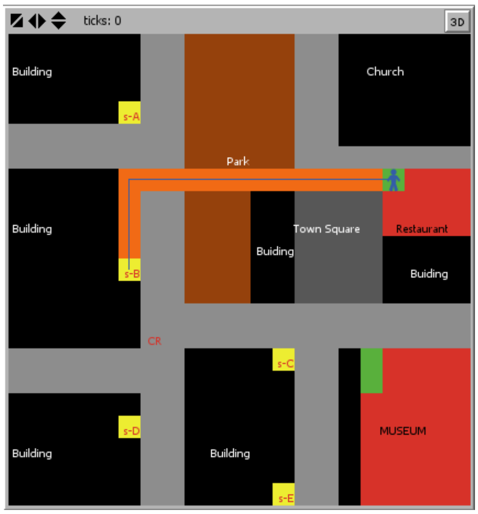

Figure 5. Screenshot of Netlogo simulation (the tourist takes a walk at the restaurant)

Rule 3a. The bus is still not at the station B but at another station and the critical time t* is reached, the tourist creates intention to get off the bus. The rule 3a is *BEL*(On (tourist, bus no 5, τ_2) ∧ ¬On (bus no 5, station B, t*) ∧ On(bus no 5, station x, t*) ∧ Finishes(t*, τ_2)) ∧*DES*(On(tourist, museum, τ_5)) → *INT*(Appear(tourist, bus no 5, t*))

Rule 3b. The bus is still not at the station B and is not at any other station (e.g. staying in traffic jam), the critical time t* has passed, the tourist creates intention to get off the bus as soon as he could t3*. The rule 3b is *BEL*(On (tourist, bus no 5, τ_2) ∧ ¬On (bus no 5, station B, t_3*) ∧ On(bus no 5, station x, t_3*) ∧ Before(t*, t_3*) ∧ Finishes(t_3*,τ_2) ∧ *DES*(On(tourist, museum, τ_5)) → *INT*(Appear(tourist, bus no 5, t_3*)); (for this situation t_3*=t_3)

Rule 3c. The bus arrives at the station B and the tourist creates intention to get off the bus i.e. appear. The rule 3c is *BEL* (On (tourist, bus no 5, τ_2) ∧ On (bus no 5, station B, t'_3) ∧ Finishes(t_3,τ_2)) ∧ *DES*(On(tourist, museum, τ_5)) → *INT*(Appear(tourist, bus no 5, t_3))

Rule 4. The tourist is at the station C and creates intention to walk toward the museum. The Rule 4 is *BEL*(On (tourist, station C, t_3) ∧ Starts(t_3, τ_3) ∧ Before(t_3, t*)) ∧ *DES*(On(tourist, museum, τ_5)) → *INT*(Closer(tourist, museum, τ_3))

Rule 5. The tourist is in front of the museum and creates intention to enter in the museum at time point t5. The rule 5 is *BEL*(FrontOf(tourist, museum, τ_4) ∧ Finishes(t_5, τ_4) ∧ Starts(t_5, τ_5)) ∧ *DES*(On(tourist, museum, τ_5)) → *INT*(On(tourist, museum, t_5))

Rule 6. The tourist is at the museum and has an intention to be at the museum during τ_5. The rule 6 is *BEL*(On(tourist, museum, τ_5) ∧ Starts(t_5, τ_5)) ∧ *DES*(On(tourist, museum, τ_5)) → *INT*(On(tourist, museum, τ_5))

Rule 7. The tourist is at the museum and has an intention to exit from museum/to be in front of the museum The rule 7 is *BEL*(On(tourist, museum, τ_5) ∧ Finishes(t_6, τ_5)) ∧ *DES*(On(tourist, restaurant, τ_8)) → *INT*(FrontOf(tourist, museum, t_6))

Rule 8. The tourist is in front of the museum and has an intention to walk toward the restaurant during τ_6. The rule 8 is *BEL*(FrontOf(tourist, museum, t_6) ∧ Starts(t_6, τ_6)) ∧ *DES*(On(tourist, restaurant, τ_8)) → *INT*(Closer(tourist, restaurant, τ_6))

Rule 9. The tourist is on the left side of the restaurant and has an intention to turn towards the restaurant. The rule 9 is *BEL*(LeftOf(tourist, restaurant, τ_7) ∧ Finishes(t_8, τ_7)) ∧ *DES*(On(tourist, restaurant, τ_8)) → *INT*(TurnRight(tourist, restaurant, t_8))

Rule 10. The tourist is in front of the restaurant and has the intention to enter the restaurant and have a lunch. The rule 10 is *BEL*(FrontOf(tourist, restaurant, t_8) ∧ Starts(t_8, τ_8)) ∧ *DES*(On(tourist, restaurant, τ_8)) → *INT*(On(tourist, restaurant, τ_8))

C. An integration process

We don't live in perfect world where everything can be predictable. Let assume that the bus no 5 (shown in Fig. 4 with circle) has been stuck in traffic jam and the tourist is still on the bus at critical time t*, which indicates that is too late for visiting the museum, so he needs to give up from original plan. (Rule 3b). When the tourist get off the bus, there is no applicable rule for new situation. Consequently, the decision generation module initiates the integration module (Fig. 1) to solve the problem. The integration module uses autonomous entities. A final result is the best path from the tourist's current position to the restaurant (Fig. 5) and new rules written by modal logic formulas are:

- *BEL*(On(tourist, station B, t^*_3)) ∧ *DES*(On(tourist, restaurant, τ^*_5)) → *INT*(TurnRight(tourist, station B, t^*_3))
- *BEL*(On(tourist, station B, t^*_4) ∧ Meets(t^*_3, t^*_4) ∧ Starts(t^*_4, τ^*_3) ∧ *DES*(On(tourist, restaurant, τ^*_5)) → *INT*(Closer(tourist, street corner, τ^*_3))
- *BEL*(On(tourist, street corner, t^*_5) ∧ Finishes(t^*_5, τ^*_3)) ∧ *DES*(On(tourist, restaurant, τ^*_5)) → *INT*(TurnRight(tourist, street corner, t^*_5))
- *BEL*(On(tourist, street corner, t^*_6) ∧ Meets(t^*_5, t^*_6) Starts(t^*_6, τ^*_4)) ∧ *DES*(On(tourist, restaurant, τ^*_5)) → *INT*(Closer(tourist, restaurant, τ^*_4))
- *BEL*(FrontOf(tourist, restaurant, t^*_7) ∧ Starts(t^*_7, τ^*_5)) ∧ *DES*(On(tourist, restaurant, τ^*_5)) → *INT*(On(tourist, restaurant, τ^*_5))

D. Simulation

The example from previous section is simulated in NetLogo (shown in Fig. 4 and Fig. 5), a multiagent programmable environment. Since originally NetLogo does not support BDI agents, two libraries that extend basic NetLogo platform with BDI features (allowing the implementation of BDI agents) are used [27]. In the simulation, elements of the environment are discrete, EBDI agents and autonomous entities can move in four directions (up, down, forward and backward). A modal formula Closer is interpreted in simulator as multiple modal formula Forward. We assumed that around every building is a sidewalk.

VI. CONCLUSION AND FUTURE WORK

In this paper the idea of integration of an agent with mental attitudes and autonomous entities are given. Each type has advantages and disadvantages [19]. We have therefore combined these type of agents making a synergy from their differences, thereby creating human-like agents with extension that enables computational efficient solution in unpredictable situation.

In future work a model of multiagent dynamic system and a hybrid multiagent system architecture with different types of agents (BDI, EBDI and autonomous entities) will be developed. Since they have different cognitive levels, a different type of interaction needs to be established. In addition, the integration module within an extended BDI agent will be expanded so it can use services of autonomous entities for searching the solution, not only for preparing an environment for searching algorithms.

REFERENCES

[1] M. Fisher, R. H. Bordini, B. Hirsch, P. Torroni, "Computational logics and agents: a road map of current technologies and future trends", Computational Intelligence, vol. 10, no.1, pp. 41-66, 2007

[2] J. Dix, M. Fisher, "Where logic and agents meet", Journal of Annals of Mathematics and Artificial Intelligence, vol. 61, no. 1, pp. 15-28, 2011

[3] W. van der Hoek, M. Wooldridge, "Logics for Multiagent Systems", AI Magazine, vol. 33, no.3, pp. 92-105, 2012

[4] M. E. Bratman, "Intentions, plans and practical reason", Harvard University Press, Cambridge, MA, 1987

[5] A. S. Rao, M. Georgeff, "Modeling rational agents within a BDI-architecture", in R.Fikes & E.Sandewall Eds. Proc. of knowledge representation and reasoning (KR&R-91), San Mateo, CA: Morgan Kaufmann, 1991, pp. 473-484.

[6] A. S. Rao and M. P. Georgeff, "BDI Agents: from theory to practice", in Proceedings of 1st International Conference on Multi-Agent Systems, pp. 312–319, 1995

[7] E. Norling, E, "Folk Psychology for Human Modelling: Extending the BDI Paradigm", in Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2004, pp. 202-209, 2004

[8] L. Padgham, D. Scerri, G. Jayatilleke, S. Hickmott, "Integrating BDI reasoning into agent based modeling and simulation" in Proceedings of the Winter Simulation Conference, pp. 345-356, 2011

[9] S. Lee, Y-J. Son, "Dynamic Learning in Human Decision Behavior for Evacuation Scenarios under BDI Framework", in Proceedings of the 2009 INFORMS Simulation Society Research Workshop, pp. 96-100), 2009

[10] K. Cho, N. Iketani, M. Kikuchi, K. Nishimura, H. Hayashi, M. Hattori, "BDI Model-Based Crowd Simulation" in Proceedings of 8th International Conference, IVA 2008, Tokyo, Japan, pp. 364-371, 2008

[11] R. Evertsz, M. Pedrotti, P. Busetta, H. Acar, "Populating VBS2 with Realistic Virtual Actors" in Proceedings of the 18th Conference on Behavior Representation in Modeling and Simulation, pp. 1-8, 2009

[12] J. Bajo, D. I. Tapia, A. de Luis, S. Rodriguez, J.F. de Paz, J.M. Corchado, "Hybrid Architecture for a Reasoning Planner Agent" in Lecture Notes in Computer Science, pp. 461-468, 2007

[13] D.I. Tapia, J. Bajo, J.M. Corchado, S. Rodriguez, J.M.Manzano, "Hybrid Agents Based Architecture on Automated Dynamic Environments", in Lecture Notes in Computer Science, pp. 453-460, 2007

[14] M. Navarro, S. Heras, V. Julian, V. Botti, "Incorporating temporal-bounded CBR techniques in real-time agents", Expert Systems with Applications, vol. 38, no. 3, pp. 2783–2796, 2011

[15] M. Navarro, J.F. de Paz, V. Julian, S. Rodriguez, J. Bajo, J.M. Corchado, "Temporal bounded reasoning in a dynamic case based planning agent for industrial environments", Expert Systems with Applications, vol. 39, no. 9, pp. 7887–7894, 2012

[16] G.I. Simari, S.D. Parsons, "Markov Decision Processes and the Belief-Desire-Intention, Bridging the Gap for Autonomous Agents", Springer, 2011

[17] Y. Chen, J. Hong, W. Liu, L. Godo, C. Sierra, M. Loughlin, "Incorporating PGMs into a BDI Architecture", 16th Intl. Conf. on Principles and Practice of Multi-Agent Systems, PRIMA 2013, pp. 54-69, 2013

[18] A. Tan, Y. Ong, A. Tapanuj, "A hybrid agent architecture integrating desire, intention and reinforcement learning", Expert Systems with Applications, vol. 38, pp. 8477-8487, 2011

[19] H. Van Dyke Parunak, P. Nielsen, S. Brueckner, "Hybrid Multi-agent Systems: Integrating Swarming and BDI Agents", 4th International Workshop Engineering Self-Organising Systems, ESOA 2006, pp. 1-14, 2006

[20] E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence: From Natural to Artificial System", New York: Oxford University Press, 1999

[21] M. Dorigo, T. Stützle, “Ant colony optimization: overview and recent advances” In Handbook of metaheuristics, Springer US, pp. 227-263, 2010

[22] S. Russell, P. Norvig, “Artificial Intelligence: A Modern Approach”, 3rd edition, Prentice Hall, 2009

[23] J. K. Allen, "Maintaining knowledge about temporal intervals", Communications of the ACM, vol. 26. pp. 832-843, 1983.

[24] L. Maleš, B. Žarnić, “A spatiotemporal model of events within a BDI”, EUROCON Int. Conf. on Computer as a tool, pp.1-4, 2011

[25] J. Liu, X. Jin, K. Tsui, “Autonomy Oriented Computing - From Problem Solving to Complex Systems Modeling”, Kluwer, 2005

[26] H. Choset, “Coverage for robotics–A survey of recent results. Annals of mathematics and artificial intelligence”, vol. 31, pp.113-126, 2001

[27] I. Sakellariou, P. Kefalas, I. Stamatopoulou, “MAS coursework design in NetLogo” in Proceeding of 8th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2009, pp. 47-54