An Agent Architecture Coupling Cognition and Emotions for Simulation of Complex Systems

Mathieu Bourgais, Patrick Taillandier, Laurent Vercouter

An Agent Architecture Coupling Cognition and Emotions for Simulation of Complex Systems

Mathieu Bourgais[1,2], Patrick Taillandier[2] and Laurent Vercouter[1]

[1] Normandie Univ, INSA Rouen, UNIHAVRE, UNIROUEN, LITIS, 76000 Rouen, France
[2] Normandie Univ, UNICAEN, UNIHAVRE, UNIROUEN, CNRS, IDEES, 76000 Rouen, France

Abstract. Nowadays, multi-agent simulations are often used to study complex systems, in particular in social sciences. However, most of the time, the developed models propose a simplistic modeling of the behavior of the different actors because of the lack of simple tools allowing the development of more complex agents. In order to fill this lack, this paper presents a cognitive agent architecture coupled with an emotional engine usable through a modeling language easy to understand by non computer scientists. This ease of use is illustrated through an example dealing with the evacuation of a city.

Keywords: Cognitive Agent, Emotions, Multi-Agent Simulation, Evacuation

1 Introduction

These last years, multi-agent simulation has become an important tool, especially in the field of social sciences [15]. It enables to simulate situations with humans at a large scale more easily than with traditional simulations.

To ensure the quality of results obtained with digital simulations, these ones have to reproduce correctly the actors' behavior, that implies not only the use of a cognitive architecture [5, 15] but also to take into account their emotions [20, 3]. However, using such a cognitive and emotional architecture requires a high level in computer sciences and more particularly in Artificial Intelligence. Moreover, as the studied systems are often composed of thousands of entities, the computation time of the proposed tools is a true concern.

This paper proposes an agent architecture integrating at the same time a cognitive and an emotional dimension, that aims at being used by all types of user - even non computer scientists - and at simulating several thousands of agents. This architecture is implemented in GAMA [1, 14], a multi-agent software for modeling and simulating complex systems. The present work consists in upgrading the existing cognitive architecture in GAMA [11] and in integrating an engine that will simulate emotions in the agents.

This article is structured as follow: Section 2 is a state of the art on the existing cognitive and emotional architectures in the field of multi-agent simulation. Section 3 presents the cognitive architecture. Section 4 is dedicated to the presentation of the developed emotional module. Section 5 discusses an example of use of the new architecture and at last Section 6 serves as a conclusion.

2 Related Works

2.1 Cognitive multi-agent simulations

A lot of cognitive architectures have been presented in the field of multi-agent simulations. Indeed, it has been shown that adding cognition to agents increases the realism of simulations [5, 15].

One of the most used architecture in this domain is the BDI architecture [10]. It has been developed using modal logic [12] to define without ambiguity the concepts of beliefs, desires and intentions and the logical links between these concepts.

The BDI architecture has been implemented in different frameworks to ease the development of cognitive agents. A classical framework is the Procedural Reasoning System (PRS) [19] that is composed of three steps : a perception that gives information about the environment to the agent, a central interpreter that deliberates between desires and finally the execution of the action selected. Inspired by PRS, other frameworks have been developed such as JACK [16] that has been used in different commercial use (in video games for example). JADE [8] is another framework enabling the development of BDI agents in multi-agent systems whose most advanced version is Jadex [22]. Unfortunately, these frameworks are developed for a public expert in computer science.

The BDI architecture has also been included in multi-agent modelization and simulation software in order to be more accessible. An extension of NetLogo [29] proposes a simplified BDI architecture for educational purposes [24]. Agents have beliefs, intentions and ways to answer to intentions. Sing and Padgham [26] went further by connecting a multi-agent platform with an existing BDI framework (JACK or Jadex for example). In the same spirit, an application connecting the Matsim platform [6] with the GORITE BDI framework [23] has been proposed. These works, more generic, are also developed for a public expert in computer science.

Another integration of the BDI architecture has been initiated in the GAMA software [1]. The developed architectures have been created to be complete as most as possible and available to a non expert public. It gives agents a beliefs base, a desires base, an intentions base and a reasoning engine. It also offers low computation time enabling the simulation of thousands of agents. The limit of this work is that it does not provide tools to connect the cognitive engine with the environment of the agent. The present article is based on this later work.

2.2 Emotional multi-agent simulations

In parallel with propositions of cognitive architectures, researches were lead to add emotions to agents [13, 3]. Various works have shown that the addition of emotions to a multi-agent system permitted to increase the realism of observed results [20, 15].

Emotional theories have been developed first by James and Lange [18] who worked on a physiologic origin of emotions. Their theory explains that the human nervous system creates a physiological answer to events perceived. Emotions are then the answers created physiologically.

Afterwards, researches have been lead to show the cognitive origin of emotions. Arnold [4] and Lazarus [27] discussed the fact that it is the appraisal of a situation that creates emotion. Over that psychological definition, Ortony, Clore and Collins [21] defined a typology of emotions (the OCC typology) based on the appraisal of different situations. This model represents twenty emotions created to be used in artificial intelligence.

Different implementations of emotions have then been proposed for agents. DETT (Disposition, Emotion, Trigger, Tendency) [28] is based on the OCC model by considering the perception of a situation as the trigger of the creation of emotions. Other works such as eBDI [17] integrated emotions to a BDI architecture. This emotional module creates primary emotions from perception of the environment and secondary emotions are created by cognition. These emotions are then used to influence the cognitive deliberation with a control loop to minimize the impact of a selected action on the emotion base. However, these integrations are not made easy to use for a non expert public.

Finally, various works have been done on the study of emotional contagion. The emotional contagion is the process wherein the emotions of a person influence the emotions of an other person or a group of persons [7]. As explained in the theoretical studies of Bosse [9] and Barsade [7], emotional contagion is a social influence. Therefore, it is important to integrate it to the present work to enable modelers to socialize cognitive agents. In their work, they revealed the existence of two types of emotional contagion: an unconscious form and a conscious form. These theories have been partially integrated in a cognitive architecture based on eBDI [25] by adding a decreasing intensity to emotions and an unconscious contagion. However, this work does not implement the conscious contagion.

3 Presentation of the cognitive module

Our architecture is based on the one developed previously for GAMA [11, 14], that aims at letting modelers use the BDI theory to define their cognitive agents. It also fosters the ease of use for a non expert public and a low computation time. In this section, we propose an extension for this architecture that links cognitive agents with their environment. The cognitive architecture with our improvements is summarized in figure 1.

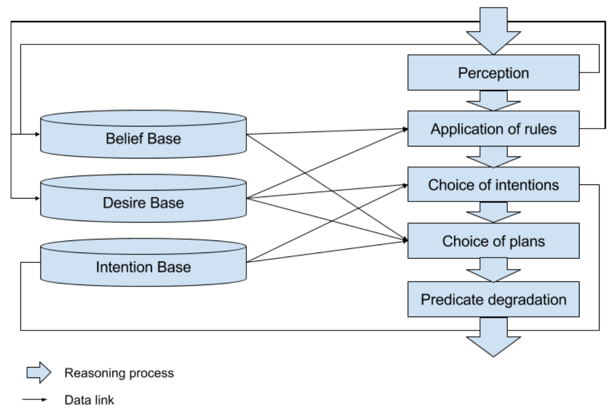

Fig. 1. Schema of our cognitive architecture

3.1 Basis of the cognitive architecture

In this architecture, the knowledge is represented by a data type called predicate. We consider a predicate to be a tuple <Name, Values, Priority, Truth> with the following elements :

- **Name** (mandatory) : the name of the predicate.
- **Values** (optional) : a map containing the values stored by the predicate.
- **Priority** (optional) : a real priority value to compare two predicates.
- **Truth** (optional) : a boolean value that indicates if the predicate is true or false.

Depending on the context, a predicate can be a belief, a desire or an intention, and can be used in the reasoning steps of choosing the current intention and choosing the current plan. These steps follow the principles of the BDI theory that is to say an intention is chosen in function of the beliefs, desires and intentions of the agent and then a plan is selected to answer this intention according to the agent's beliefs. The reasoning engine is detailed in the paper [11].

This architecture gives a cognitive behavior to agents but does not link this behavior with the agent's environment. This is the challenge of adding a perception, inference rules and a lifetime to predicates.

3.2 Tools to connect the architecture to the environment

The perception module creates beliefs representing the perceptions of the agent. It contains multiple parameters to be adaptable to various situations :

- A perception area around the agent. The modeler can give a particular geometry (it can be complex and defined according to the vision of the agent) or just a perception distance.

– The agents that will be possible to perceive. The modeler can indicate precisely which agents will be detected in the area of perception.
– A boolean condition to activate the perception.

Then, the modeler indicates which information of the perceived agents will become beliefs for the perceiving agent.

The definition of inference rules consists in parametrizing the transformation of beliefs and desires according to the perceived world. A rule can add or delete beliefs or desires according to the state of the corresponding bases. The inference rules get a lot of parameters, all optional, to be adjustable to any situation :

– A belief to activate the rule.
– A desire to activate the rule.
– The belief created by the rule.
– The desire created by the rule.
– The belief deleted by the rule.
– The desire deleted by the rule.
– A boolean condition to activate the rule.

Finally, a lifetime is added to predicates which become a tuple <Name, Values, Priority, Truth, Lifetime>. It is optional and decreases automatically of 1 at each step with the predicate degradation module of the figure 1. This mechanism enables to indicate that a belief or a desire will not exist any longer after a certain number of time steps.

With these tools, the cognitive architecture gets its own dynamic and is connected to the environment of the agent. An experiment has been carried out with three computer scientists and three geographers who were invited to test the architecture through an exercise. The six modelers said the architecture was easy to use which is a good point. The details of the experiment can be find on the site : http ://acteur-anr.fr/experiment.html

4 Adding emotion through cognition

4.1 Presentation of the emotion type

We have chosen to base the emotion module on the OCC theory [21]. The main contribution of the present work consists in the implementation of this emotion system in the core of our cognitive architecture in order to propose to modelers a tool to easily create emotional agents.

According to the OCC model, an emotion is a valued answer to the appraisal of a situation. On that statement, we create the emotion type as a tuple <Name, Predicate, Intensity, Decay> with the following elements :

– **Name** (mandatory): the name of the emotion.
– **Predicate** (optional): the predicate, identified by its name, that represents the fact about which the emotion is expressed.

– **Intensity** (optional): a real positive value of the intensity of the emotion.
– **Decay** (optional): a real positive value which will be subtracted from the intensity value at the end of each time step.

So, for example, an agent can simply be happy (which corresponds to the tuple <"happy">), an agent can express hope about a fact A (which corresponds to the tuple <"hope", A>) or an agent can express fear about a fact B with an intensity of 3.8 and a decay value of 1.2 (which corresponds to the tuple <"fear", B, 3.8, 1.2>).

We add to the cognitive architecture of GAMA an emotion base that will contain instances of the emotion type. In this base, only one emotion with a same name and a same predicate can exist. So, if an existing emotion is added again to the base, the emotion base only keeps one emotion and, adds the two intensity values and keeps the minimum decay value so that an ephemeral emotion does not put to end a more sustainable emotion.

4.2 Creation of emotions by and for cognition

To take into account emotions in the behavior of agents, we modified the cognitive architecture as described by figure 2.

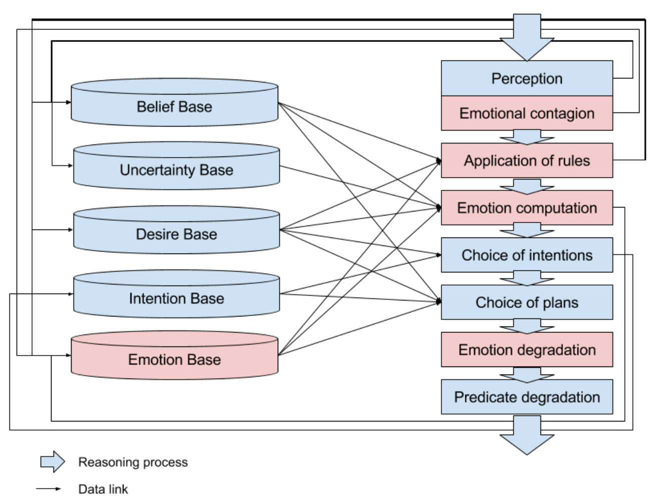

Fig. 2. Schema of the developed architecture with the addition of emotions

To integrate emotions into the cognitive architecture, in addition to the emotion base mentioned previously, we add a base of uncertain expected events that will be referred thereafter as the uncertainty base. The main contribution of this article is about the emotion base which will be managed with different tools.

Concerning the uncertainty base, it is for the moment managed manually by the modeler.

To manage the emotion base, we modify the inference rules previously defined in order to take into account emotions. The modeler can, thanks to the definition of those rules, change the emotion base according to the agent mental states or change the belief base or desire base according to the agent emotions.

The use of inference rules to create emotions from cognition requires knowledge in emotions theory from the modeler. We also offer for non expert researchers the possibility to use emotions with the computation emotion module presented in figure 2. This module will automatically, with no intervention from the modeler, create emotions according to the mental states of the agent.

The automatic creation of emotions according to the mental states of the agent is based on the OCC model [21] and its logical formalism [2], which has been proposed to integrate the OCC model in a BDI architecture.

According to the OCC theory, emotions can be split into 3 groups: emotions linked to events, emotions linked to people and emotions linked to things. For the moment, we only focused on emotions linked to events that are the easiest to integrate in a generic way as the concept of events is represented in the existing architecture with the type predicate.

The eight emotions automatically created by our computation module have the following definition, according to the logical formalism [2]:

- **Joy(P)** <”joy”,P> = Belief(P) & Desire(P)
- **Sadness(P)** <”sadness”,P> = Belief(P) & Desire(not P)
- **Hope(P)** <”hope”,P> = Expect(P) & Desire(P)
- **Fear(P)** <”fear”,P> = Expect(P) & Desire(not P)
- **Satisfaction(P)** <”satisfaction”,P> = Hope(P) & Belief(P)
- **Disappointment(P)** <”disappointment”,P> = Hope(P) & Belief(not P)
- **Relief(P)** <”relief”,P> = Fear(P) & Belief(not P)
- **Fear confirmed(P)** <”fear_confirmed”,P> = Fear(P) & Belief(P)

The terms Belief(P), Desire(P) and Expect(P) have the following definitions:

- **Belief(P)** : indicates that the fact P belongs to the belief base of the agent.
- **Desire(P)** : indicates that the fact P belongs to the desire base of the agent.
- **Expect(P)** : indicates that the fact P belongs to the uncertainty base of the agent.

The emotions created by the computation module are created without intensity nor decay value as there is no generic way to define them. If the modeler wants to use intensity, he/she will have to add it manually.

Note that the creation of some emotions will automatically remove other ones. The creation of fear confirmed or of relief will replace the emotion of fear while the creation of satisfaction or of disappointment will replace a hope emotion. Moreover, according to the logical formalism [2], the creation of satisfaction and relief leads to the creation of joy and the creation of disappointment and fear confirmed leads to creation of sadness.

Once the decisional process of the agent has been carried out, the architecture deteriorates the emotions according to the decay value of each emotion. The operation done here is only a subtraction that corresponds to the degradation of an intensity of an emotion during one time step. If an intensity reaches zero, the emotion is removed from the base.

With these tools, the agent can now express emotions according to its mental states, it can use these emotions to influence its deliberations and these emotions can dynamically evolve with their decay rate.

4.3 Emotional contagion module

In our architecture, emotional contagion is seen as an emotional perception. The emotional contagion is thus defined as the process wherein the emotions of an agent are influenced by the emotions of agents nearby. The module of emotional contagion is then placed inside the perception module, as seen on the figure 2, to actualize the emotion base before starting the reasoning phase.

The unconscious emotional contagion corresponds to the fact that an emotion is automatically copied when perceived in another agent. In our work, the unconscious emotional contagion has multiple parameters to be adaptable on various situations:

- The emotion that must be detected to be copied (mandatory).
- A real value of charisma between 0 and 1 corresponding to the strength of the transmission from the perceived agent (optional).
- A real value of receptivity between 0 and 1 corresponding to the capacity to receive an emotion for the perceiving agent (optional).
- A real threshold between 0 an 1. If the product between charisma and receptivity is greater or equal to the threshold, the emotional contagion is activated (optional).
- A boolean value to activate the emotional contagion (optional).

By default, the charisma value and the receptivity value is 1 and the threshold is 0.25 in order that if the value of the two variables are 0.5, the contagion happens. With that system, if the charisma is just under a mean value (under 0.5), the receptivity will have to be above the mean value, and vice versa. If the modeler wants another threshold, he/she can change it easily.

Let's take an example. An agent A have a charisma value of 0.5 and a receptivity value of 0.6 and an agent B have a charisma value of 0.7 and a receptivity value of 0.2. The threshold for the emotional contagion stay by default at 0.25. If agent A is contaminated by agent B about an emotion E, the emotional contagion will happen ($0.7 * 0.6 \geq 0.25$). Agent A will copy emotion E and the intensity of E will be multiplied by the charisma value from B and the receptivity value from A. In the case of a contamination of B from A, the contagion will not happen ($0.5 * 0.2 < 0.25$).

The conscious emotional contagion corresponds to the fact that an agent creates a new emotion if she perceives a certain emotion in another agent. The created emotion can be different from the perceived emotion, contrary to the unconscious emotional contagion. The conscious emotional contagion has also multiple parameters to be adaptable to various situations:

- The emotion that must be detected to activate the contagion (mandatory).
- The emotion that will be created by the contagion (mandatory).
- A real value of charisma between 0 and 1 (optional).
- A real value of receptivity between 0 and 1 (optional).
- A real threshold between 0 an 1 (optional).
- A boolean value to activate the emotional contagion (optional).

5 Example case

5.1 Introduction of the example case

The architecture developed in this work has been used on the example case of drivers in a crisis situation such as an industrial hazard.

The situation is the following one: on the studied area, a factory of chemical products is on fire and some products are spread into the air. If a driver smells the toxic gas, she will think it is possible that a catastrophe happened so she will go to a shelter. Some drivers will not be afraid by the gas but when they will see the fire, they will panic and go to the shelters twice faster than the normal speed.

Over that situation, we add the fact that if a driver sees other drivers fleeing, she will have a probability to flee herself.

This example uses at the same time the cognitive architecture and the emotional engine to simulate the behavior of thousands of agents.

5.2 Implementation of the example

We focus on the implementation of the behavior of drivers. Each driver gets a random value for charisma and receptivity and a boolean value that indicates if the driver will flee only with the fear of a catastrophe (and not necessarily with the belief of a catastrophe). Initially, the driver agents desire to move randomly.

The reasoning cycle of the agent starts with the perception of her own environment. An example of implementation of the environmental perception of the cloud gas is given in Figure 3. The modeler indicates the target that will be perceived and a distance for that perception and he/she indicates that if the "hazard" agent is perceived, the driver agent should add the uncertainty about a catastrophe and the desire that there is no catastrophe.

On the other side, the perception of other drivers is an emotional perception that will occur through emotional contagion. The figure 4 shows an example of implementation for the unconscious contagion of the fear confirmed emotion.

We then define inference rules to create the agent's desires. Figure 5 shows the implementation of a rule to remove the intention to move randomly and to add the desire to go to a shelter if the agent has the belief of a catastrophe.

```
perceive target:hazard in: hazard_distance{
    ask myself {
        do add_uncertainty(catastrophe);
        do add_desire(nonCatastrophe);
    }
}
```

Fig. 3. Implementation of the perception of the cloud gas

```
perceive target:people in: other_distance{
    unconscious_contagion emotion:fearConfirmed;
}
```

Fig. 4. Implementation of the emotional contagion

```
rule belief:new_predicate("catastrophe")
    remove_intention:at_target new_desire:in_shelter;
```

Fig. 5. Definition of an inference rule

Finally we implement plans that will be used by the agent to face hers intentions. On the figure 6, the activation of the plan is conditioned by the presence in the emotion base of the fear confirmed emotion.

```
plan evacuation intention: in_shelter emotion: fearConfirmed
```

Fig. 6. Implementation of a plan

The complete model can be find on the following website :
https://github.com/mathieuBourgais/ExampleModels.

5.3 Discussion

The goal of this example is not to propose an accurate representation of drivers' behavior in a crisis situation but to show how the tools we have developed can be used for such a simulation.

The major challenge of our architecture is to be easy to use, especially for non computer scientists. The codes shown as examples illustrate the simplicity of implementation for the perceptions, inference rules or emotional contagion.

One of the main advantage of our work is its emotion computation module. On the previous example, the modeler never needs to indicate how to create emotions. They will be automatically created according to the cognitive state of the agent. The modeler just has to indicate the changes of the cognitive states such as adding beliefs or uncertainties. Then, the modeler can use the emotions inside inference rules or plans.

Another advantage of this architecture is its adaptability. A more skilled modeler can, for example, use charisma and receptivity values to modulate the emotional contagion or create new emotions using inference rules. The architecture proposes a lot of tools to easily create complex behaviors for agents.

Concerning the computation time, the experiments carried whith our example on a computer with a dual core processor and 4GB of RAM showed that it takes less than 50ms per step for 500 driver agents, which is quite fast. With 5000 agents, it takes less than 400 ms per each step which is an acceptable time.

6 Conclusion

In this article, we have presented a cognitive architecture for multi-agent simulations coupled with an emotional module enabling the modeler to define agents with social features. Our architecture features tools to connect agents' cognition and their environment such as perception or inference rules, an emotion type and an emotion engine to automatically create emotions from cognition, and an emotional contagion module.

In the future, we plan to lead experiments with modelers of different fields to test the ease of use of the architecture. We also want to facilitate the use of the uncertainty base in the cognition process. Finally, in order to enrich the possibilities of developping social agents, we plan to add social norms to the architecture while keeping in mind the low computation time and the ease of use.

Acknowledgment

This work is part of the ACTEUR ("Spatial Cognitive Agents for Urban Dynamics and Risk Studies") research project funded by the French National Research Agency.

References

1. Gama website, 2016.
2. Carole Adam. Emotions: from psychological theories to logical formalization and implementation in a bdi agent, 2007.
3. Carole Adam, Richard Canal, Benoit Gaudou, Ho Tuong Vinh, Patrick Taillandier, et al. Simulation of the emotion dynamics in a group of agents in an evacuation situation. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 2010.
4. Magda B Arnold. Emotion and personality. 1960.
5. Tina Balke and Nigel Gilbert. How do agents make decisions? a survey. *Journal of Artificial Societies and Social Simulation*, 2014.
6. Michael Balmer, Marcel Rieser, Konrad Meister, David Charypar, Nicolas Lefebvre, Kai Nagel, and K Axhausen. Matsim-t: Architecture and simulation times. *Multi-agent systems for traffic and transportation engineering*, 2009.
7. Sigal G Barsade. The ripple effect: Emotional contagion and its influence on group behavior. *Administrative Science Quarterly*, 2002.

8. Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. Jade–a fipa-compliant agent framework. In *Proceedings of PAAM*. London, 1999.
9. Tibor Bosse, Rob Duell, Zulfiqar A Memon, Jan Treur, and C Natalie Van Der Wal. A multi-agent model for emotion contagion spirals integrated within a supporting ambient agent model. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 2009.
10. M Bratman. Intentions, plans, and practical reason. 1987.
11. Philippe Caillou, Benoit Gaudou, Arnaud Grignard, Chi Quang Truong, and Patrick Taillandier. A simple-to-use bdi architecture for agent-based modeling and simulation. In *ESSA 2015*, 2015.
12. P. R. Cohen and H. J. Levesque. Intention is choice with commitment. 1990.
13. Jonathan Gratch and Stacy Marsella. A domain-independent framework for modeling emotion. *Cognitive Systems Research*, 2004.
14. Arnaud Grignard, Patrick Taillandier, Benoit Gaudou, Duc An Vo, Nghi Quang Huynh, and Alexis Drogoul. Gama 1.6: Advancing the art of complex agent-based modeling and simulation. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 2013.
15. S. Hickmott, C. Adam, B. Gaudou, and D. Scerri. Agents bdi et simulations sociales, unis pour le meilleur et pour le pire. 2011.
16. Nick Howden, Ralph Rönnquist, Andrew Hodgson, and Andrew Lucas. Jack intelligent agents-summary of an agent infrastructure. In *5th International conference on autonomous agents*, 2001.
17. Hong Jiang, Jose M Vidal, and Michael N Huhns. Ebdi: an architecture for emotional agents. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. ACM, 2007.
18. Carl Georg Lange and William James. *The emotions*. Williams & Wilkins, 1922.
19. Karen L Myers. User guide for the procedural reasoning system. *SRI International AI Center Technical Report.*, 1997.
20. Ranjit Nair, Milind Tambe, and Stacy Marsella. The role of emotions in multiagent teamwork. *Who Needs Emotions*, 2005.
21. Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 1990.
22. Alexander Pokahr, Lars Braubach, and Winfried Lamersdorf. Jadex: A bdi reasoning engine. In *Multi-agent programming*. Springer, 2005.
23. Ralph Rönnquist. The goal oriented teams (gorite) framework. In *International Workshop on Programming Multi-Agent Systems*. Springer, 2007.
24. Ilias Sakellariou, Petros Kefalas, and Ioanna Stamatopoulou. Enhancing netlogo to simulate bdi communicating agents. In *Hellenic Conference on Artificial Intelligence*. Springer, 2008.
25. Julien Saunier and Hazaël Jones. Une architecture mixte agent/environnement pour le calcul dynamique des émotions. In *JFSMA*, 2012.
26. Dhirendra Singh and Lin Padgham. Opensim: A framework for integrating agent-based models and simulation components. In *Frontiers in Artificial Intelligence and Applications-Volume 263: ECAI 2014*. IOS Press, 2014.
27. Craig A Smith and Richard S Lazarus. Emotion and adaptation. 1990.
28. H Van Dyke Parunak, Robert Bisson, Sven Brueckner, Robert Matthews, and John Sauter. A model of emotions for situated agents. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006.
29. Uri Wilensky and I Evanston. Netlogo: Center for connected learning and computer-based modeling. *Northwestern Univ., Evanston, IL*, 1999.