Integrating Expectations into Jason for Appraisal in Emotion Modeling

Joaquín Taverner, Bexy Alfonso, Emilio Vivancos and Vicente Botti
Departmento de Sistemas Informáticos y Computación, Universitat Politècnica de València,
Camino de Vera s/n, 46022, Valencia, Spain

Keywords: Agents, Emotion Modeling, Appraisal, Expectations, Jason.

Abstract: Emotions have a strong influence on human reasoning and behavior, thus, in order to build intelligent agents which simulate human behavior, it is necessary to consider emotions. Expectations are one of the bases for emotion generation through the appraisal process. In this work we have extended the Jason agent language and platform for handling expectations. Unlike other approaches focused on expectations handling, we have modified the agent reasoning cycle to manage expectations, avoiding complex additional mechanisms such as monitors. This tool is part of the *GenIA[3]* architecture and, hence, is a step towards the standardization of the emotion modeling process in BDI (Beliefs-Desires-Intentions) agents.

1 INTRODUCTION

Human behavior is not merely rational. It is influenced by individuals' emotional characteristics such as personality, emotions or mood. Psychological and neurological sciences support this statement, offering the foundation for the research on this field (Damásio, 1994; Baumeister et al., 2007). In recent years several studies have been proposed for modeling emotions using software processes and software agents. When software agents need to show a human-like behavior they need to include emotions in their reasoning process. However not many proposals in this field offer general practical tools oriented to the design and implementation phase.

A key process in emotion modeling is the appraisal process. In psychology, appraisal theories argue that emotions are the result of the interpretations and explanations that each individual performs based on his/her circumstances and concerns. In this evaluation process, an individual interprets his/her relationship with the environment (Scherer, 2001; Lazarus, 1994). As part of this, judgments are performed according to certain criteria or variables called appraisal variables. A set of specific emotions result from the appraisal process corresponding to different configurations of these judgments (Lazarus, 1994). There are several criteria for evaluating an event that takes place in the individual environment. One of the most accepted criteria is the implications of the event in the future. This criteria may result in emotions such as hope or fear in relation to things that will happen, or may result in emotions that are reactions to violations of expectations (Winkielman et al., 1997; Castelfranchi and Lorini, 2003) such as surprise or disappointment. Our approach is focused on this kind of emotions. For computationally modeling the appraisal of an event in relation to its future implications, it is necessary to evaluate the likelihood of the event consequences (Golub et al., 2009), as well as the value of the unexpected and the possibility of change. Therefore, it may be necessary to count on a representation of future expectations. Mechanisms to evaluate the probability of events, actions and their consequences may be also required.

According to the OCC model (Ortony et al., 1988), a widely accepted model of emotions that offers a mechanism of appraisal and a classification of 22 emotion types, one of the appraisal variables that affects almost all emotion types is related to expectancy, because expectations affect all emotional experiences. This is also supported by most appraisal theories like (Scherer, 2001; Roseman, 2001; Lazarus, 1994). Expectations are an anticipatory mental component (Ranathunga et al., 2011), which allow individuals to satisfy their need of anticipating future events in order to blur the unexpectedness of an unknown situation.

In order to allow defining expectations in an agent program we have extended the syntax of the Jason agent language (Bordini et al., 2007), allowing to define the probability and a time interval for expectations that may be used in the appraisal process of an affective agent. In our approach expectations can

Taverner, J., Alfonso, B., Vivancos, E. and Botti, V.
Integrating Expectations into Jason for Appraisal in Emotion Modeling.
DOI: 10.5220/0006070202310238
In *Proceedings of the 8th International Joint Conference on Computational Intelligence (IJCCI 2016) - Volume 1: ECTA*, pages 231-238
ISBN: 978-989-758-201-1

be defined in a similar way to beliefs. We have also created a method for keeping track of fulfilled and not fulfilled expectations. Besides, in order to show how expectations can indirectly influence an agent decisions, we use an example where agents expectations influence the agent affective state, and in turn, this affective state influence further decisions. This extension is part of the development of the *GenIA*[3] architecture (Alfonso et al., 2014; Alfonso et al., 2016) presented in the section 2.1.

The rest of the paper is organized as follow. Section 2 presents the supporting affective theories and the *GenIA*[3] architecture. The method proposed for defining and monitoring expectations is presented in Section 3. An example of how to use expectations in a Jason agent that plays the BlackJack game is presented in Section 4. Finally, a comparative study with other similar approach is showed in Section 5, and Section 6 offers some conclusions.

2 BACKGROUND AND SUPPORTING THEORIES

When computer scientists model affect, they face two broad challenges: how to model affect and how to enrich artificial agents architectures and languages to include those affective models (Reisenzein et al., 2013). Several psychological theories provide almost complete support for affect-related processes (e.g., emotion generation, and emotions effects on cognition, expression, and behavior) (Hudlicka, 2014). Consequently, approaches for agent modeling are different considering what psychological theories support them (Reisenzein et al., 2013). For a review about computational approaches for modeling emotions see (Marsella et al., 2010), and (Reisenzein et al., 2013).

Nevertheless, the task of systematically recreating existing emotion theories, and to use the general strategy of building formal languages for this, may be cumbersome. Two more viable strategies can be used: “1) break up existing emotion theories into their component assumptions and 2) reformulate these assumptions in a common conceptual framework”(Reisenzein et al., 2013). In (Alfonso et al., 2014; Alfonso et al., 2016) it is proposed *GenIA*[3], which is a common conceptual framework that supports different emotion theories.

2.1 The *GenIA*[3] Architecture

GenIA[3](a General-purpose Intelligent Affective Agent Architecture) is a BDI (Beliefs-Desires-Intentions) agent architecture designed to create affective agents. This architecture is grounded on general aspects of widely accepted psychological and neurological theories but it doesn’t makes any commitment with any particular theory. Nevertheless it offers a default design that includes an appraisal process inspired by (Marsella and Gratch, 2009).

GenIA[3] includes all the processes of a BDI agent as well as a set of main affective processes according to a solid theoretical background (see Figure 1). According to *GenIA*[3] these are the processes that should be considered when building appraisal-based models for affective agents (Alfonso et al., 2014; Hudlicka, 2014; Marsella et al., 2010). As part of the BDI processes, it includes a belief revision function (**brf**), which determines new beliefs starting from a perceptual input and the agent’s current beliefs; an options generation process (**options**), which takes the agent’s current beliefs and intentions to determine its desires (options or courses of actions available), i.e. the means to achieve its intentions; a filter process (**filter**), which determines the agent’s intentions, i.e. what to do, through a deliberation process that uses previously-held intentions, and the agent’s current beliefs and desires (the new set of intentions will contain either newly adopted or previously-held intentions); and action selection function (**execute**), which returns the next action to be executed on the basis of current intentions. The affective processes, on the other hand, include an **appraisal** process, whereby an evaluation of the current situation is performed considering the agent environment, cognitive state, and concerns, and where a set of appraisal variables are derived ; **affect generator**, where the appraisal variables that result from the appraisal process are transformed into a representation of the agent’s affective state; **affect regulator**, which determines the possible emotional behaviors and coping responses for the given situation; **affective modulator of beliefs** which determines if and how the affective state biases the agent’s beliefs, contributing to the beliefs maintenance according to the affective state; and the **affect’s temporal dynamic**, which doesn’t depend of any other process and no other process depends on it, and determines the duration of the affective state’s components as well as how their intensities decay over time.

3 PROPOSAL

In this section we present the proposed method for defining and managing expectations. In order to allow an agent programmer to define expectations we have extended the Jason language including a new structure for expectations that can have associated a probabil-

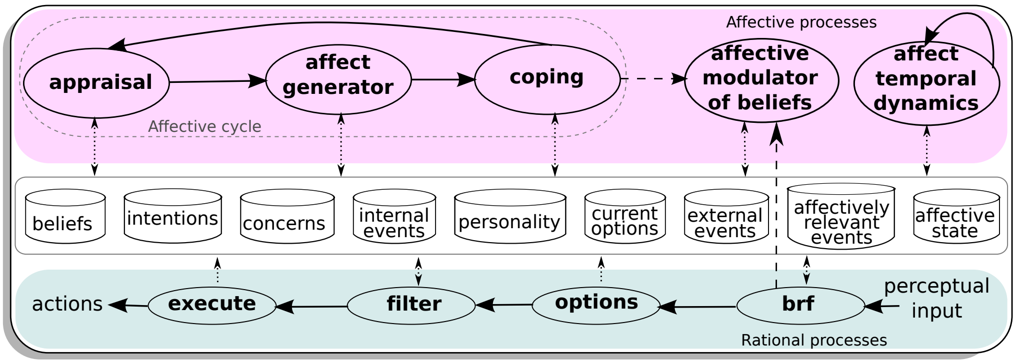

Figure 1: *GenIA*[3]: a General-purpose Intelligent Affective Agent Architecture that integrates BDI and affective processes. Sequences are represented as solid line arrows, subprocess as dashed line arrows, and exchange of information as dotted line arrows.

ity value and a time range. Also, in order to identify the influence of the expectations in the agent affective state, we allow to define the valence associated with the fulfillment of an expectation, (i.e. `positive` or `negative`). The expectations management, in turn, is performed at certain points of the agent BDI reasoning cycle. To this end, we have included a new step in this reasoning cycle. Next we briefly introduce Jason and its characteristics as an agent language (Section 3.1). We also semi formally present the extension of the Jason language (Section 3.2) and the extension of its reasoning cycle (Section 3.3).

3.1 Jason Agent Language

Jason (Bordini et al., 2007) is an interpreter for an extended version of AgentSpeak. Jason agents are BDI based, thus agents continuously decide actions to perform to reach their goals. A Jason agent program can contain beliefs, goals, and plans. Beliefs represent the knowledge the agent has about its environment, himself or about other agents. Goals represent the agent desires, and plans consist of a sequence of actions that are designed to either reaching those goals or simply respond to events. Events in Jason are generated by the addition or deletion of beliefs, the addition or failure of goals, or the addition or failure of test goals[1]

On the other hand the agent life cycle is controlled by the reasoning cycle (RC) that is continuously executed (see Figure 2).

According to the Jason semantic, several steps are performed in an agent RC. The initial step is in charge of processing the received messages (`ProcMsg`). Next an event is selected to be processed in the `SelEv` step. Then, in the step `RelPl`, a set of relevant plans are selected, and in the step `ApplPl` a subset of this set is

[1]Test goals are those goals that aim to retrieve information from the agent belief base.

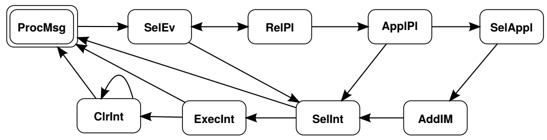

Figure 2: The reasoning cycle of a Jason agent (Bordini et al., 2007).

beliefs → ((literal|literal_prob) ".")*
literal_prob → ["~"] atomic_form_prob
atomic_form_prob → (**<ATOM>** | **<VAR>**)
["(" list_of_terms ")"]
["[" list_of_terms "]"]
[t_point_range]
t_point_range → "<" arithm_expr "," arithm_expr ">"

Figure 3: Simplified extension of the EBNF for including expectations in the Jason Language.

selected, which contains the applicable plans[2]. Next a plan is selected for its execution in the step `SelAppl`; this implies the addition of an intended mean (or a new intention) what is performed in the step `AddIM`. Finally, a particular intention is selected for its execution from the set of intentions (`SelInt`), and the cycle ends by clearing the finished intentions (`ClrInt`).

3.2 Extension of the Jason Language

We have modified the syntax of the Jason language to allow the definition of expectations with a probability and a time range. Figure 3 shows a part of the Jason language EBNF. The complete EBNF of the Jason language can be found at (Vieira et al., 2007).

Figure 3 shows that the rule for defining beliefs has been extended so that it can be defined

[2]Relevant plans are those whose triggering event matches the event being processes, and applicable plans are those whose conditions for being executed are fulfilled according to the agent beliefs.

as a `literal` according to its original definition or as a `literal_prob`. When written according to `literal_prob`, a belief becomes an *expectation*. According to the definition of `literal_prob` it is possible to associate a time range to an expectation, which must follow the structure of `t_point_range`. A time range allows expectations to be time-bound, so that it is possible to hypothesize about something for a period of time in the present and/or future. Thus a time range includes an initial time and a final time. The initial time can be expressed either as an arithmetical expression that can include the reserved word 'Now', or directly the word 'Now'. We have created this word 'Now' for easily referring to the "current time" (in milliseconds). The final time can be expressed either as an arithmetical expression or as the reserved word 'Infinite' (indicating an undetermined time in the future). Moreover, a belief can also become an expectation if one of its annotations is a probability. A probability is one annotation with the form `prob(P,V)`, where `P` is a numerical value between 0 and 1 and `V` is an optional component that can have the values `positive` or `negative`. We have included this component to simulate what an appraisal process would do in order to use expectations for determining their influence in the agent affective state. For example consider the next portion of a Jason code:

```
belief[prob__(Number,Connotation)]<T1, T2>
```

In this example the expectation has a structure similar to the structure of a belief, that has a probability `Number`, a valence `Connotation` and a time range (`T1` represents the initial time and `T2` represents the final time). We use the probability to indicate the level of expectedness of the expectation. Thus we can use this probability to check the impact generated by an expectation in the calculation of the affective state (Golub et al., 2009). We also allow to specify the valence of expectations to indicate whether the consequences of their fulfillment are positive or negative. Determining the valence of expectations may be one of the tasks performed by an appraisal process, so that future extensions of the present approach won't need the expectations' valence component. Finally, one of the innovations we propose is the possibility of defining expectations for a time range. Within this time range the expectations can be fulfilled. Once the time range has ended, it is considered that expectations haven't been fulfilled. For example, one expectation where the agent believes that the weather will be cloudy with a probability of 0.5 (at some point between now and within two hours) and where the agent considers that a cloudy weather is "something good" can be written as:

```
time(cloudy)[prob__(0.5,positive)]<Now,
Now+2*60*60*1000>
```

If within that time range the `time(cloudy)` belief is inserted in the agent belief base (either through the perception process or by a message received from another agent), then the expectation will be fulfilled. If two hours later no belief `time(cloudy)` is perceived or received as a message, the expectation is considered not fulfilled.

3.3 New Step in the Jason Reasoning Cycle

A Jason agent configuration is defined by a tuple $\langle ag, C, M, T, s\rangle$ (Vieira et al., 2007). The components of this tuple can be modified on each step of the agent reasoning cycle. The first component (*ag*) represents the agent program which contains a set of beliefs *bs* and a set of plans *ps*. *C* represents the agent circumstance, containing the current set of intentions, events, and actions to be performed in the agent environment. *M* is the component that stores the agent communication aspects. *T* stores temporary information including relevant plans in relation to an event (*R*), applicable plans (Ap), and data considered in a particular reasoning cycle including the current intention (ι), event (ε), and applicable plan (ρ). Finally *s* contains the step of the reasoning cycle being executed, where $s \in \{$*ProcMsg, SelEv, RelPl, ApplPl, SelAppl, AddIM, SelInt, ExecInt, ClrInt*$\}$. In our approach we have included a new element *es* in the agent program *ag*, representing the set of expectations of the agent. Also, the agent temporary information *T* has been modified. We have included a new component *Exp* in *T* that represents the expectations pending of being processed by any dependent process such as an appraisal process. We define *Exp* as a tuple $\langle fpe, fne, nfpe, nfne\rangle$ were fpe represents the set of fulfilled positive expectations, fne the set of fulfilled negative expectations, $nfpe$ the set of not fulfilled positive expectations, and $nfne$ the set of not fulfilled negative expectations.

One expectation can be removed due to three reasons: (1) because there is an action in the agent code to eliminate it, (2) because it has been fulfilled or (3) because it hasn't been fulfilled. In the last two cases, we keep a record of this expectation. This record is able to differentiate fulfilled from unfulfilled expectations and also positive from negative (consequences of) expectations (fulfillment). We use this differentiation for determining the influence on the affective state. Following this criteria *Exp* is updated in two different moments during the agent reasoning cycle. Firstly it is updated in the brf (belief revision function), in charge of updating the agent's beliefs according to what is perceived. Secondly it is updated at

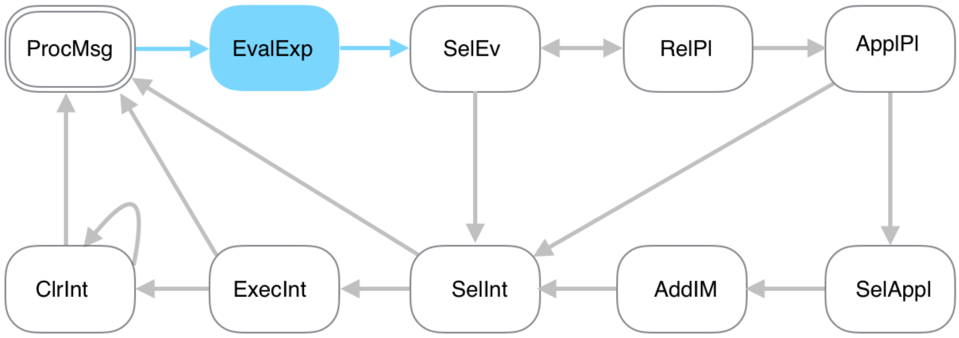

Figure 4: The Jason agent reasoning cycle extended with a new state *EvalExp*.

a fixed point of the agent reasoning cycle. We have modified the brf so that, besides adding and removing beliefs, expectations are also added and removed. Every time a belief is inserted we check if that belief fulfills some expectation. But as an expectation's time range may not be active at the time of the insertion or removal of a belief, we need a way of constantly evaluate expectations. In order to evaluate expectations in a consistent way we have modified the Jason reasoning cycle by including a new step *EvalExp* as shown in Figure 4.

The second moment for updating Exp is performed in the `EvalExp` step. This step has been introduced after `ProcMsg` and before `SelEv`, so that the transition from `ProcMsg` to `SelEv` has been removed and two new transitions from `ProcMsg` to `EvalExp` and from `EvalExp` to `SelEv` have been created. This modification implies that, after starting the reasoning cycle (`ProcMsg`) the next step is `EvalExp`, where expectations are evaluated and updated according to the transitions rule EvalExp[3].

$$\frac{Exp = (fpe, fne, nfpe, nfne)}{\langle ag, C, M, T, \texttt{EvalExp}\rangle \rightarrow \langle ag, C, M, T', \texttt{SelEv}\rangle} \quad \text{(EvalExp)}$$

$$\begin{aligned}
\text{where: } T'_{Exp_{fpe}} &= EvFPE(Exp) \cup fpe \\
T'_{Exp_{fne}} &= EvFNE(Exp) \cup fne \\
T'_{Exp_{nfpe}} &= EvNFPE(Exp) \cup nfpe \\
T'_{Exp_{nfne}} &= EvNFNE(Exp) \cup nfne
\end{aligned}$$

All new functions introduced in the transition rule presented determine a set of expectations with the form (b, p, v, t_i, t_f), where b is the expectation, p the probability value, v the expectation valence (`positive` or `negative`), t_i the initial time for the expectation, and t_f the final time. Given the set bs of agent beliefs, the set es of the agent expectations, and the current time ct, the previous functions are defined as follows:

Definition 1.

$$\begin{aligned}
EvFPE(es) = \{&(b, p, v, t_i, t_f) | b \in es, b \in bs, \\
&t_i >= ct, ct <= t_f, v = \text{`}positive'\}
\end{aligned}$$

Definition 2.

$$\begin{aligned}
EvFNE(es) = \{&(b, p, v, t_i, t_f) | b \in es, b \in bs, \\
&t_i >= ct, ct <= t_f, v = \text{`}negative'\}
\end{aligned}$$

Definition 3.

$$\begin{aligned}
EvNFPE(es) = \{&(b, p, v, t_i, t_f) | b \in es, b \notin bs, \\
&t_f > ct, v = \text{`}positive'\}
\end{aligned}$$

Definition 4.

$$\begin{aligned}
EvNFNE(es) = \{&(b, p, v, t_i, t_f) | b \in es, b \notin bs, \\
&t_f > ct, v = \text{`}negative'\}
\end{aligned}$$

4 EXAMPLE

To demonstrate how to use expectations in Jason agents, we have developed a game based on the BlackJack cards game (Griffin, 1986). In this game the player must get cards with a total value of twenty one for reaching a blackjack. These type of games are very useful for simulating decision under the influence of emotions, because it has been widely studied the role that emotions play in decision-making in these contexts. Besides, games allow to see changes in the emotional state, as players usually are happy when they win and get angry when they lose.

We have created two agents as shown on Figures 5 and 6. On the upper side it is shown a player (an affective agent). On the bottom side, it is shown the 'Bank' which is a Jason agent without emotions and completely rational. The agent player starts the game by

[3] We have followed the same notation as (Vieira et al., 2007) where attributes are represented as a subindex. For example, the beliefs set bs of the agent program ag is represented as ag_{bs}

Figure 5: Graphical Interface. Board situation for the player (top) and the bank (bottom). The player loses.

Figure 6: Graphical Interface. Board situation for the player (top) and the bank (bottom). The player wins.

asking a card. Depending on its points and its affective state it must choose between stand or hit. Figure 5 shows a situation where the player loses and Figure 6 shows a situation where the player wins. With this example we have shown that expectations are crucial for generating emotions since, only by modeling expectation it has been possible to simulate a variety of situations completely coherent with what would happen in the real life. It should be clear that this is just a simple example for the purpose of showing how expectations could be used to determine their influence on the affective state and to determine how this affective state may impact decisions. We have represented the affective state through the emoticons shown on Figure 7.

Figure 7: Emoticons that represent the different affective states (depressed, sad, neutral, satisfied, happy).

As stated before, the affective agent is able to decide according to its affective state. To do this, we have included a new annotation in the label of the agent plan. This new annotation has the form `[affect__(Number)]`, where `Number` indicates the value of the current affective state. We have modified the function that selects the plan to be executed to consider the current affective state.

```
@p2[affect__(2)]
+card(Value)[source(house)] : myPoints(N)
<-
    -card(Value)[source(house)];
    -+myPoints(N+Value);
    !nextPlay(17.0, N+Value).
```

For example, in the previous fragment of code there is a plan that will be executed if the affective state is 2 (sad). The triggering event of this plan is `+card(Value)`. This event represents that it has got a new card, and the variable `N` represents the total points of the player before the new card. As part of the actions of the plan, the agent number of points is updated by adding the value of the new card. Also as part of the plan, the agent creates a new goal `nextPlay(17.0,N+Value)`, where 17.0 represents the points for standing, and `N+Value` is the total points the affective agent has.

In this example, the affective state is only modified by using the evaluations (positive or negative) of the expectations, as well as the information about whether they were fulfilled or not. The affective state is a value that can have five possible ranges of values one per emoticon of Figure 7: depressed, sad, neutral, satisfied, and happy respectively. Following this example, the affective agent has the expectation:

```
win(round)[prob__(0.5,positive)]<Now,Now+10000>
```

According to this expectation the agent is waiting to win this round[4] with a probability of 0.5. The agent also has an expectation where: it is waiting to have twenty one at some point in the game, it is waiting to win two consecutive rounds at some point in the game, and it is waiting to get a card with 10.0 points in the current round:

```
get(21.0)[prob__(0.1,positive)] <Now, Infinite>
win_followed(2)[prob__(0.6,positive)] <Now,
```

[4] In our proposal, we consider that the rounds shall last for ten seconds

```
Infinite>
get(10.0)[prob__(0.3,positive)] <Now, Now+10000>
```

All expectations are positive, so when any of them is fulfilled, this will have a positive effect on the agent affective state. However, if expectations are not fulfilled, this will have a negative effect on the agent affective state. To check the impact that the fulfillment or not fulfillment of expectations has in the affective state we use the probability. Expectations with a high probability will have less impact (exactly the complement of the expectation probability) than those with a low probability. For example, if the player gets a card with a value of 10.0 units the affective state will be increased in 0.7, otherwise the affective state will be decreased in 0.3 units.

5 RELATED WORK

There are some precedents in incorporating expectations at Jason agents (Cranefield, 2014). Probably the approach most similar to ours is the one proposed in (Ranathunga et al., 2010; Ranathunga et al., 2011; Ranathunga and Cranefield, 2012). The authors incorporate a mechanism for expectations management through monitors. Similar to ours, this approach is able to detect expectations fulfillment and violation at execution time. This allows to monitor changes in the environment and the behavior of others to this end. Nevertheless, in this approach expectation fulfillment is notified through event generation, allowing to handle those events through plans. By contrast in our approach expectations are used as appraisal variables which can be used to determine emotions generated, and may modify the agent affective state.

In Ranathunga's approach an expectation is active if an associate condition is fulfilled. Also there is no time management for expectations, what is detrimental to the expectation expressivity. In our approach expectations can be define for being fulfilled or violated at a time range, thus an expectation becomes active when its time range becomes active. Unlike Ranathunga's approach, were there is a "separate mechanism for obtaining information from the system in which the Jason agent is situated" (Ranathunga et al., 2011), in our approach information from the system is obtained through the own Jason mechanisms and tools. This makes the sensoring and perception process more transparent to the user.

Finally in Ranathunga's approach, unlike ours, it is not possible to express to what extent the agent expects something (i.e. it is not possible to assign a probability to expectations). Also, monitors are used to make the evaluation processes while we use the own reasoning cycle of the agent. Therefore our proposal avoids the costs associated with the concurrence derived from the use of monitors. Our approach also keeps a record of fulfilled and not fulfilled expectations for being used and managed by the process that may need them, such as the appraisal process.

6 CONCLUSIONS

Emotions play an important role in the decision-making process of humans. Currently tools for allowing the implementation of affective agents are scarce. As discussed in this paper, expectations handling is the base for simulating prospective emotion generation through the appraisal process. We have extended Jason to introduce expectations representation and handling in agents as a part of the development of the *GenIA³* architecture.

The new proposed method for expectations handling is fully integrated into a BDI agent reasoning cycle, so that no additional structures (what may compromise the agent performance) were needed. Also this method is able, not only to detect the fulfillment or not fulfillment of expectations within a time range, but also to keep track of this. This record can be used for determining the influence of expectations on the affective state of the agent. The example used shows that the probability associated to the expectations may have a great importance on determining the impact of the fulfillment or not of an expectation on the affective state as well. Clearly, variations on an individual affective state are not only determined by expectations, nevertheless they are necessary to determine this influence. We have confirmed this by observing a player agent playing a cards game with a feedback of its affective state. The use of an affective agent as an artificial player clearly showed good chances in order to improve the game experience.

As part of our future work we will improve the default implementation of *GenIA³* by creating new methods for determining the values of other important appraisal variables such as "desirability" or "controllability". Also, more sophisticated methods for determining how these appraisal variables impact the agent affective state, and how this affective state influences the decision making process will be included.

REFERENCES

Alfonso, B., Vivancos, E., and Botti, V. (2016). Towards Formal Modeling of Affective Agents in a BDI Architecture. *ACM Transactions on Internet Technology*, To appear.

Alfonso, B., Vivancos, E., and Botti, V. J. (2014). An Open Architecture for Affective Traits in a BDI Agent. In *Proceedings of the 6th ECTA 2014. Part of the 6th IJCCI 2014*, pages 320–325.

Baumeister, R. F., Vohs, K. D., DeWall, C. N., and Zhang, L. (2007). How emotion shapes behavior: Feedback, anticipation, and reflection, rather than direct causation. *Personality and Social Psychology Review*, 11(2):167–203.

Bordini, R. H., Hübner, J. F., and Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason*. Wiley.

Castelfranchi, C. and Lorini, E. (2003). Cognitive anatomy and functions of expectations. *Proceedings of IJCAI'03 Workshop on Cognitive Modeling of Agents and Multi-Agent iterations*.

Cranefield, S. (2014). *Agents and Expectations*, pages 234–255. Springer International Publishing, Cham.

Damásio, A. R. (1994). *Descartes' error: emotion, reason, and the human brain*. Quill.

Golub, S. A., Gilbert, D. T., and Wilson, T. D. (2009). Anticipating one's troubles: the costs and benefits of negative expectations. *Emotion*, 9(2):277–281.

Griffin, P. (1986). *The Theory of Blackjack: The Compleat Card Counter's Guide to the Casino Game of Twenty-one*. Faculty Publishing.

Hudlicka, E. (2014). From Habits to Standards: Towards Systematic Design of Emotion Models and Affective Architectures. In *Emotion Modeling*, pages 3–23. Springer International Publishing.

Lazarus, R. (1994). *Emotion and Adaptation*. Oxford University Press.

Marsella, S. C. and Gratch, J. (2009). EMA: A process model of appraisal dynamics. *Cognitive Systems Research*, 10(1):70–90.

Marsella, S. C., Gratch, J., and Petta, P. (2010). Computational models of emotion. In *A Blueprint for Affective Computing: A Sourcebook and Manual*, Affective Science, pages 21–46. OUP Oxford.

Ortony, A., Clore, G. L., and Collins, A. (1988). The cognitive structure of emotions. *Cambridge University Press, Cambridge, MA*.

Ranathunga, S. and Cranefield, S. (2012). Expectation and complex event handling in bdi-based intelligent virtual agents (demonstration). In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, pages 1491–1492.

Ranathunga, S., Purvis, M., and Cranefield, S. (2010). Integrating expectation monitoring into jason: A case study using second life. *Presented at the 8th European Workshop on Multi-Agent Systems.*

Ranathunga, S., Purvis, M., and Cranefield, S. (2011). Integrating expectation handling into jason. *The Information Science*, (2011/02).

Reisenzein, R., Hudlicka, E., Dastani, M., Gratch, J., Hindriks, K., Lorini, E., and Meyer, J.-J. (2013). Computational modeling of emotion: Toward improving the inter-and intradisciplinary exchange. *IEEE Transactions on Affective Computing*, 4(3):246–266.

Roseman, I. J. (2001). *A Model of Appraisal in the Emotion System: Integrating Theory, Research, and Applications*, pages 68–91. Oxford University Press.

Scherer, K. R. (2001). Appraisal considered as a process of multilevel sequential checking. *Appraisal processes in emotion: Theory, methods, research*, 92:120.

Vieira, R., Moreira, Á. F., Wooldridge, M., and Bordini, R. H. (2007). On the formal semantics of speech-act based communication in an agent-oriented programming language. *J. Artif. Intell. Res.(JAIR)*, 29:221–267.

Winkielman, P., Zajonc, R. B., and Schwarz, N. (1997). Subliminal affective priming resists attributional interventions. *Cognition and Emotion*, 11(4):433–465.