DE GRUYTER OPEN
RIGA TECHNICAL UNIVERSITY
Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2017, vol. 22, pp. 5–12
doi: 10.1515/acss-2017-0012
https://www.degruyter.com/view/j/acss

General Guidelines for Design of Affective Multi-Agent Systems

Māra Pudāne[1], Egons Lavendelis[2]
[1, 2] *Riga Technical University, Latvia*

***Abstract* – The paper presents general guidelines for designing affective multi-agent systems (affective MASs). The guidelines aim at extending the existing agent-oriented software engineering (AOSE) methodologies to enable them to design affective MASs. The reason why affective mechanisms need specific attention during the design is the fact that the way how both rational tasks and interactions are done differ based on the affective state of the agents. Thus, the paper extends the traditional design approaches with the design of affective mechanisms and includes them in the design of the system as a whole.**

***Keywords* – Affective computing, affective multi-agent system, agent-oriented software engineering, multi-agent system design.**

I. Introduction

Multi-agent paradigm has been recognised for implementing intelligent autonomous and distributed behaviour. The paradigm has been there already for several decades. Still, agent-oriented software engineering (AOSE) is not among the mainstream software development approaches used by the industry. In the last years of the 20th century and the first decade of the 21st century it was thought that the most important missing part was methodological support [1], [2]. Despite the fact that many methodologies have been proposed during the past 20 years [3], there is still no wide use of agent paradigm in industrial projects. Many agent-oriented developments do not report that they have used any software development methodology [4]. The authors of the paper have identified the following reasons: the range of developed methodologies is very wide and at the same time the choice of the methodology depends on the system. Therefore, the development teams have to learn various methodologies. One approach to deal with this problem is to develop a unified AOSE methodology. Some efforts in this direction can be found in [5], [3]. Additionally, the methodologies do not include any support for designing additional, system specific parts. At the same time, there is a very wide range of agents. The unified methodology should be capable of engineering all the corresponding aspects of the system.

One of the areas extending the notion of multi-agent systems is affective computing, which is a growing research direction that aims at giving an ability to perceive, understand, feel and express emotions to artificial systems. One type of such systems is agent communities that can share affects as well as interact with a user and amongst themselves by using emotions – affective multi-agent systems (affective MAS). In affective MASs, the affective part of agents makes them to be a specific class of MASs that need appropriate tools for design. The authors find that natural implementation of affects can become a more significant reason to adopt the multi-agent paradigm in the industry because of the abilities of affective MASs to implement not only intelligent and autonomous behaviour, but also fully human-like reasoning and behaviour in terms of emotions.

From the design perspective, the main difference in affective MAS design is in the fact that affective mechanisms are influencing how other (rational) tasks are done by agents. Affects should be treated as states that can completely change the behaviour of agents rather than executable functions. There are also a variety of affective abilities that various agents might possess, starting form full-fledged emotions to following simple affective patterns. Additionally, the communication between autonomous agents is influenced by the affective state of the participants of the negotiations. As a consequence, neither tasks nor interactions can be designed independently of the affective mechanisms in the agents. The paper takes the existing AOSE methodologies as methodological background and provides guidelines how to introduce design of affective components into the traditional AOSE process.

The remainder of the paper is organised as follows. Section II gives a general description of design phase in the existing AOSE methodologies. Section III outlines the existing design principles of affective computing systems. Section IV describes the proposed guidelines while Section V concludes the paper.

II. Design of MAS

At the beginning of the 21st century, MAS was recognised as a ready for use paradigm that could theoretically contribute to solving various distributed problems. At that moment, complexity of distributed autonomous system development was considered to be the main obstacle for MASs to penetrate software development industry [6]. To overcome this obstacle, many efforts have been devoted to the methodological perspective of MAS development and development tools. As a result, several dozens of agent-oriented software engineering methodologies have been developed so far. These methodologies are usually based on the existing things in similar domains. As a consequence, according to [2] the existing methodologies can be divided into the following groups by their origin: extended object-oriented methodologies, extended knowledge engineering methodologies, organisation design based methodologies and newly developed methodologies from the perspective of intelligent systems.

Well-known examples of AOSE methodologies based on object-oriented software engineering methodologies are MaSE, O-MaSE, PASSI, INGENIAS and ADELFE 2.0. Gaia 2, Prometheus and MASSIVE belong to a class of newly developed or artificial intelligence based methodologies. MAS-CommonKADS and Tropos are the best-known examples of knowledge engineering based methodologies. Styx and ARCHON Framework are well known in the class of organisation design based AOSE methodologies [1], [3], [6].

Depending on the class of the methodology, the techniques used vary widely. Still the software engineering process includes phases originally defined in the waterfall model, namely, analysis, design, implementation, testing, deployment and maintenance. Although the process is iterative and does not comply with the waterfall process, these phases can be used to analyse the techniques corresponding to a particular phase. The scope of the paper is limited to the design phase and, therefore, the remainder of the section concentrates on this phase.

Almost all of the methodologies defined above explicitly divide the design phase into design of macro level and micro level [1]. Although the names differ between methodologies, during the first stage or macro level design, a multi-agent system is designed, while during the second stage individual agents are designed.

The boundaries between phases are the following. The analysis phase answers the question "what does the system have to do?" All steps and techniques related to this question are included in the analysis phase. The design phase answers the question "How will the system achieve the functionality defined during the analysis phase?" The macro level designs everything that is related to more than one agent while the micro level designs details of each individual agent.

In addition to the macro and micro level design, the deployment of a particular system is designed. It is specified which particular instances of each agent class are deployed. For each instance a platform where it is deployed and various parameters are specified. One example of deployment diagram is proposed in MaSE methodology [7].

A. Macro Level Design

Macro level design is the first stage of design phase. It takes the requirements specification as an input and defines which agents will be implemented in the system and how they will interact. The two most important parts of the macro level design are agent definition and interaction design.

Depending on the actual AOSE methodology, an agent definition can include the following tasks:

- task definition. Requirements are transformed into tasks (sometimes named functionalities) that must be done by agents. Some methodologies ([8], [2]) do it already in the analysis phase, while a majority include them in the design phase. In this paper, it will be considered as part of design phase since it is influenced by the design of affective mechanisms;
- role definition. Tasks are grouped into roles. During this step, similar tasks are grouped together. As a result, roles are defined. A role is a coherent group of functionality that is fulfilled by the same agent;
- agent definition. Roles are grouped into agent classes (sometimes named types). During this step, agent classes are defined in terms of the roles that they fulfil. Some methodologies omit roles and immediately define agents based on the set of tasks.

The agent definition usually is a straightforward process when all three steps are done one after another. After finishing this process, the next step is interaction design. During this step, interactions among agents are defined. Interaction design can be done at three levels. Depending on the methodology and the system under design, either one or even multiple of these three levels can be used [2]:

1. The acquaintance level can be designed to identify which pairs of agents interact. This form of interaction modelling is useful to evaluate complexity of the design, but is insufficient as a final interaction design of the system.
2. Interaction design can be done in the form of messages sent among agents. This level includes all details of interactions, but does not specify the context of messages. This is the most convenient form of interaction design [9] in case of a low number of messages.
3. Protocol design is the most detailed form of interaction design. Usually it is done in the form of AUML protocol diagram standardised by FIPA [10]. In addition to the second level of interaction design, the context and order of messages are defined. At each moment of interaction, all possible choices how to respond are specified. This form of interaction design specifies all possible details.

In many cases, interaction and protocol design are used together. To combine them, first, the interaction level is designed. During this process, in many cases the diagram becomes hard to read because of many messages among particular agents and unclear relations among messages. In such cases, messages belonging to the same conversation are substituted with one link denoting a protocol and the particular messages are moved into the protocol diagram. This approach is followed by the MASITS methodology [2], while the Prometheus methodology leaves only protocol names on the interaction level and designs particular messages only at the protocol diagram [8].

In addition to the interaction design, a common knowledge structure is designed. Interaction design and knowledge structure design are done in parallel, iteratively designing interactions and extending the knowledge structure with the concepts used in interactions, for example [9]. The aim of the knowledge structure definition step is to define a common structure for the whole system and ensure that all agents understand the concepts used in the same way. Usual form for knowledge structure is ontology. FIPA has standardised the use of ontologies in agent communication [11]. Designing the knowledge structure according to these standards enables the usage of existing automated mechanisms for encoding and decoding knowledge from message contents.

B. Micro Level Design

The task of the micro level design is to specify how the functionality and interactions defined for each agent will be implemented. Each agent is designed separately based on its inputs and outputs. One of the techniques used ([2], [8]) is to take the model of the system and to "zoom in" until the level of a particular agent. As a result, all the inputs and outputs of the agent are available and the designer's task is to design the internal structure of the agent. Agents can have the following inputs: incoming messages (sometimes grouped into protocols), events occurred in the system, data read from some repository as well as user inputs, while the outputs are messages sent (sometimes grouped into protocols), tasks done, including writing data into some repositories and giving output to the user.

This stage of the design strongly depends on the implementation platform. The concepts and methods used depend on the type of the agents used in particular platforms. Three groups of approaches can be identified here:

1. Designing deliberative agents according to the BDI agent platforms, such as JACK and JADEX [12]. In this case, the agents are designed based on the human-like BDI concepts of beliefs, desires and intentions. In practical reasoning systems, these concepts are extended with notions of event and plan. Agents can also be decomposed into components called capabilities. In this case, each capability is designed in the same way as the whole agent.
2. Designing reactive agents according to the platforms that concentrate on middleware development, for example JADE. One example is MASITS methodology that designs behaviour based agents for JADE platform [9].
3. Performing only high-level design in general concepts that can be implemented in any platform. It is followed by Gaia methodology [13]. The main advantage is the fact that the methodology can be applied to different types of agents and implementation platforms. However, practical implementation becomes complex because the design must be first transformed from the concepts used in the design process to the concepts used in a particular agent development platform.

The above-mentioned techniques are general enough to design affective MASs, still in addition to the traditional things that are covered in the AOSE methodologies, development of affective MAS requires designing one more complex part of the system – the affects. For the micro level the paper will follow the third approach with only high-level design to create an approach that can be used for different types of agents or even heterogeneous MASs because agents in affective MASs can be very different based on the affective capabilities that they must have, starting from simple reflex agents and ending with extended BDI agents whose reasoning is also affected by emotions.

None of the AOSE methodologies currently supports implementation of affects. As a consequence, it is proposed to follow the path chosen in other specific agent-oriented software engineering methodologies, in particular, to take the best practices of AOSE methodologies and combine them with the existing design methods of a particular type of systems, in this case affective software. The existing methods for designing affective software systems in the context of affective MAS are analysed in the next section and the guidelines for the design of affective MAS are presented in Section IV.

III. Design Related Issues in Affective MAS

Development of affective MAS has some additional issues that are related to affect modelling. This difference has even been marked by proposing the third class of requirements: in addition to functional and non-functional requirements, it has been proposed to have supra-functional [14] or affective [15] requirements that are aimed at adapting to users' personality in order to achieve positive results [14] and impact the way the system performs in general.

Research on design and development of affective systems is scarce. Since affective systems have not become mass production software yet, only some guidelines can be found in developing software for such systems. The vast majority of existing work has been focused on human-computer interaction [16], [17], mainly user modelling [14] and system adaptation to a user [18] as well as physical device, such as affective wearables, design [19] without general guidelines. However, affective computing has reached a level of maturity where the discussion of appropriate design and development tools and methods for internal models has started.

This can be proven with the study of Hudlicka [20], which offers a detailed description of generic affective requirement analysis framework. The framework includes 12 steps that correlate with MAS requirement analysis as well as design. The steps that correspond to the requirements analysis are: defining the functions of emotions in the system, defining specific emotions and moods, defining emotion triggers and emotional behaviours as well as defining agent embodiment. The following steps correspond to the design phase:

1. Define model's abstraction level. This step includes defining cognitive concepts, such as goals, plans, etc.
2. Select theoretical perspectives for generation and emotion effects. Here the psychological perspective is chosen, depending on the affects defined in the requirements phase. This step also includes several guidelines for choosing agents' affective architecture, i.e., if there is no model of emotions, the agent's emotions will be reactive.
3. Detail emotion triggers and emotion effects, i.e., what kind of behaviours and reasoning patterns emotions will trigger as well as what will trigger the emotions themselves.
4. Define affective dynamics that will regulate the change of emotions over time, such as intensity, magnitude, decay etc.
5. Define abstract computational tasks that were defined before to match system functions.
6. Select reasoning formalisms, such as IF..THEN rules, fuzzy logic etc.

An interesting study can be found in [15]. The authors use common software design concepts such as UML state machine diagram for design of an affective system. This is done by integrating SysML and EmotionML. Emotion ML is a structured XML-like language that allows storing emotion related information.

There have been some studies that tackle design related issues on the topic of interactions, e.g., dialogue keeping; however, these developments still examine only interaction between an affective system and a user [21].

While there are several studies in this area, they do not focus on MAS. Consequently, there are almost no guidelines on interaction, and no guidelines on what happens when multiple agents with various affective abilities are integrated in one system. Affective abilities are not the only thing that can make affective MAS heterogeneous. The emotion theories have three large groups [22], namely, categorical emotion theories, which manifest that every emotion belongs to a basic emotion category [23], dimensional theories that view emotions as a point in a multi-dimensional space (e.g., PAD, which includes three dimensions – Pleasure, Arousal and Dominance [24]), as well as appraisal theories that consider emotions to be the result of evaluation of a current state (e.g., OCC model that has 3 appraisal categories [25]). Some of these theories may be integrated into one system for various purposes, e.g., personality and mood modelling.

The authors have found that the appropriate interaction design, integration of multiple affective abilities and psychological models are the design challenges of affective MAS. To tackle these issues, modification of non-affective AOSE methodologies has been proposed in the following section.

IV. Guidelines for Design of Affective MAS

The present section aims at tackling the challenges defined in the previous section. As there are existing developments for the design of internal structure of an agent, the main changes and guidelines are directed towards the macro level with only some guidelines concerning the micro level.

A. Macro level Design of Affective MAS

Any affective system is composed of the two interrelated parts – affective and rational. While these components might not be clearly distinguishable and separable in the already implemented system, it is natural to design these parts separately and then merge them at some point of design. In case of affective MAS, there are first affective and rational tasks, then affective and rational roles which are then joined into agents.

The macro level design of affective MAS includes several additions (marked in grey in Fig. 1) to classical approaches. First, in the task definition step affective supertasks are identified. Affective supertasks represent system's affective abilities and help ensure that all intended affective requirements are designed. Affective supertasks are then broken down to more detailed tasks which can be grouped into affective roles. The design of rational tasks and roles does not differ from a classical approach. Later on, agents are defined based on rational and affective roles.

Secondly, the design of interactions includes affective interaction design. Affective interactions differ from rational interactions since the interpretation is ambiguous. As the interpretation of emotion cues lies within a receiver [26], changes concerning interpretation will be done at the micro level.

The MAS can contain a variety of agents with different affective abilities – thus, there can be multiple agents with the same or similar rational tasks but which should be implemented completely differently due to the affective processes.

Affective ability of the system must be chosen according to the requirements defined before. Although there is a possibility to define the system in free text form, the authors strongly recommend using a framework for ensuring that all affect-relevant details are collected. For this purpose, Hudlicka's first four steps for requirements analysis, identified in the previous section, should be used.

The authors propose supertasks corresponding to the four main affective processes. Hudlicka has identified two of these processes, namely, emotion generation and emotion effects on cognition and behaviour [20]. Petrovica and Pudane [27] have transformed Hudlicka's model and added emotion acquisition and emotion expression for a full affective model. Combination of these task groups allows identifying various affective abilities.

Acquirement (Ac) should be implemented if interaction with a user is required or a system will contain affective interactions among agents. It includes affective cue detection either from another agent's message or user input, i.e., events.

Expression (Ex) should be implemented if interaction with a user is required or a system will contain affective interactions among agents. Expression involves invoking of a system emotional behaviour pattern via various communication channels.

Generation (Ge) involves the interpretation of cues and applying it to the context and mapping stimuli to an emotion. Generation is needed if an agent needs to calculate an internal emotional state.

Emotion effects (Ef) is the process that includes the effects of the emotions on rational thinking and behaviour. This process differs from emotion expression, since the effects of emotion are unintentional. Some systems, such as strategy adaptation systems, skip this process by just executing some behaviour pattern (such as smiling) without having an actual effect on behaviour.

In Table I, some of the affective abilities of agents are identified with corresponding task groups. The "+" sign marks the task groups needed, "+/–" denotes the task group that may or may not be implemented to achieve goals (i.e., the variations which might be used). One of the affective abilities which is often used in the user emotion analysis, is for system to be able just to acquire emotions [28] or acquire them and generate an internal state, i.e., to interpret the reasons of emotions (often used in user modelling) [27]. Affective

tutoring systems often use just the expression of internal state according to user's responses or include generation of an affective state as well [29]. Some systems generate affective state, acquire and express, however, do not "feel" emotions, i.e., do not consider them in a behaviour and reasoning (e.g., ALMA architecture that simulates a virtual human for tutoring purposes [30] or WASABI – affective architecture that is verified on a museum guide [31]). If full user simulation or full human/entity behaviour is needed, all processes might be required [27], [32]. For training purposes, a system can be used that expresses emotions and generates an affective state that impacts its behaviour; however, does not acquire anything from a human, for example, a training system for volunteer firefighters [33].

TABLE I

SOME OF AGENT AFFECTIVE ABILITIES IN VARIOUS SYSTEMS

Agent's abilities	Ac	Ex	Ge	Ef
Ability to express an affective state		+	+/–	+/–
Ability to acquire an affective state	+		+/–	
Ability to express and acquire an affective state without generating a state	+	+		
Ability to generate an internal affective state, acquire affects, process them and generate a response	+	+	+	
Full affective mechanism	+	+	+	+

After supertasks are defined, at least one subtask for each supertask must be chosen. The guidelines here are as follows:

- Ac: different tasks for various affect sources (facial expressions, user's behaviour, body postures etc.) which are closely related to agent embodiment, (in particular, available sensors) defined in requirements task;
- Ex: different tasks for emotion expression through various modalities (face, gaze, speech, body, etc.) which are closely related to agent embodiment (in particular, effectors) defined in requirements task;
- Ge: different tasks for various affective models from the point of view of psychology and affective dynamics models that correspond to Hudlicka's design Tasks 2 and 4 listed in Section III;
- Ef: different tasks for various internal triggers and various cognitive contents as various triggers and cognitive contents are basis for choosing agent's architecture.

Then, the affective and rational tasks should be grouped into roles following these guidelines:

1. The affective tasks should be grouped among themselves only if there is no conflict with embodiment of a particular agent, i.e., there is no situation when triggers from user facial expressions are needed, yet camera is not available to an agent.
2. The affective tasks of Ge and Ef should be grouped among themselves only if there is no conflict in tasks (there is no situation when the same task is fulfilled with various means, for example, emotion decay is not calculated with exponential and linear functions or for short-term emotion modelling both OCC model and PAD space are used).
3. The affective tasks and rational tasks at this point can be merged only if a rational task is fulfilled just in one affective role to create mixed roles.

Finally, the roles can be merged into agents by joining one or several affective roles and one or several rational roles. The affective roles cannot be merged in one agent if they contain tasks from Ge superclass to avoid conflicts.

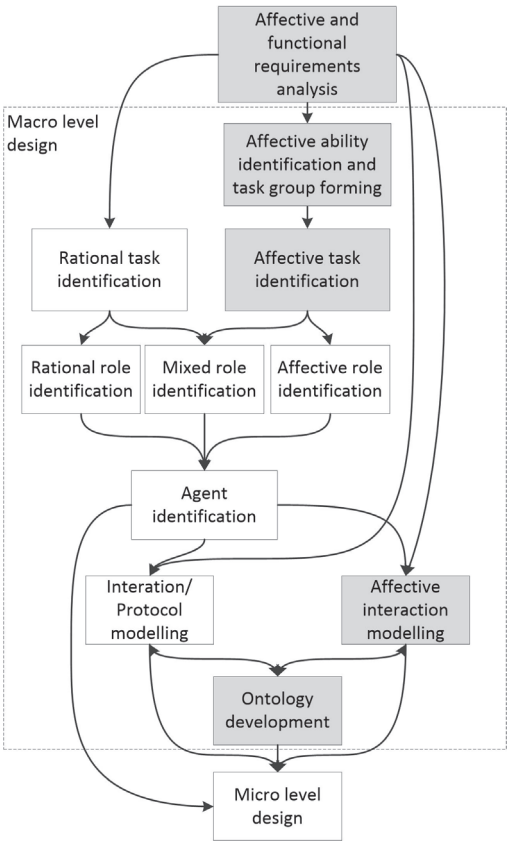

Fig. 1. Macro level design of affective MAS.

The interaction among affective agents includes not only rational interactions and agent's reactions towards them but also affective interactions, i.e., specific patterns that ensure the transfer of emotions. There are 5 possible affective interaction scenarios among agents [26] that include subconscious mechanisms from the perspective of a sender (i.e., messages that are semantically sent with no purpose of sending message) and conscious mechanisms, namely, direct communication and manipulation.

The design of subconscious mechanisms is similar to broadcast since subconsciously an agent gives his affective state to any agent to whom it is related. The subconscious messages do not have to have a response about receiving the message since it is an underlying mechanism of affective communication but it is not the main means for communication. If this is crucial for the operation of the system or for survival of agents, there might be a request to say how an agent feels (yet in this case the expression is not anymore subconscious).

The direct messages should have a reply of receiving a message; however, there are no universal guidelines for what the sequence of conversation is – thus, interaction level can be used in general for affective MAS development while in specific cases it might be needed to use a protocol level.

In parallel to the interactions, it is needed to design affective ontology. While not all affective terms might be accessible to all agents, some common knowledge of affective space is needed. For this reason, there have been ontologies developed that can be used, e.g., [34].

The following steps: supertask identification, affective task identification, affective and mixed role identification, affective interaction modelling and affective ontology modelling complement the existing macro level design approaches to help develop affective MASs. While the macro level has most changes, some of them have effect on the micro level which is discussed further.

B. Micro level Design of Affective MAS

At the micro level, it is needed to perform three tasks (Fig. 2). First, it is needed to finish the design of the emotion computation model (ECM). Secondly, appropriate architecture, reasoning mechanisms as well as interpretation mechanisms must be designed for each agent. Finally, some deployment issues should be solved.

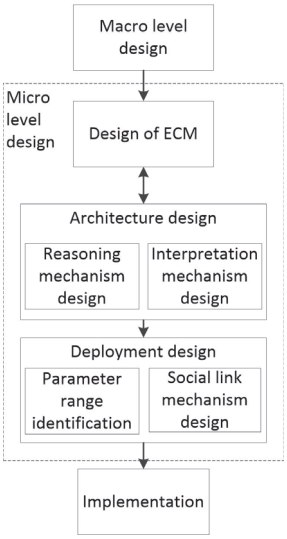

Fig. 2. Micro level design of affective MAS.

While the main tasks and components of the ECM model have been identified at the macro level, there are several computational tasks to solve, one of the most important being: how will models be transformed from one to another and integrated amongst themselves (Hudlicka's design Task 5)? Some of these transformations can be found in literature, e.g., [27] uses a formula to translate from the OCEAN personality model to PAD values.

From the architectural perspective, rational accomplishment of agent tasks fully depends on the affective model that was defined at the macro level. Depending on the affective ability, it might include agents of various reasoning mechanisms. For the architecture, recommendations are as follows:

- if an agent has only acquirement or expression related abilities and/or a psychological model contains just basic emotion groups, an agent may be implemented as a reactive agent;
- for other affective abilities and/or psychological models, deliberative agents are needed.

For the purposes of communication, it is needed to define knowledge structure that enables interpretation of received messages. Normally, this should be done by using private ontology which works as an addition to public ontology developed at the macro level design.

The last step here includes solving some design issues that concern the deployment of the model. Two things should be defined here. First, the range of parameters should be determined. Range of affective variables determines how extreme agent's affective behaviours will be. There are no specific guidelines here since it depends on the system's purpose except that there should be elaboration on this issue.

Secondly, since affective message interpretation heavily depends on social links, a mechanism that allows instances of agents to form them is needed, in other words, to conclude, whether another agent is a friend or a foe.

V. Virtual Classroom as a Case Study of the Proposed Guidelines

The guidelines were applied to design a virtual classroom aimed at teaching emotional intelligence. Student's task in the game is to play a board game. The teacher intervenes when students are being rude or emotionally unintelligent. User is one of the students in a classroom who sees other student's texts and is located in front of a computer equipped with a camera.

To design such a system, first, the requirements in correlation to Hudlicka's guidelines were defined. These requirements and their explanations are as follows:

1. *Functions of emotions in the system:* to simulate players, as an indicator that a player (either virtual or real) is not behaving appropriately.
2. *Specific emotions and moods*: user's emotions should be acquired in basic emotion groups (namely, joy, anger, sadness, fear, surprise); emotions in the game are based on appraisal of events of the game. Students should have personality based on the trait model OCEAN (includes five traits, namely, openness, consciousness, extraversion, agreeableness, and neuroticism). Emotions should decay exponentially and linearly. Emotion activation and expression should depend on personality.
3. *Emotion triggers and effects (behaviours):* triggers are an objective utility in a game of agent self and other agents, and social deduction (others emotions); effects are the changed affective state, text output when an agent achieves emotion intensity thresholds.
4. *Agent embodiment:* the system is equipped with a camera and a chat window, as well as a game window as output.

After requirement definition, *task groups or supertasks* should be defined. Since the system aims at full affective mechanism simulation, all four supertasks should be chosen: Ac, Ex, Ge, and Ef.

Then, according to requirements, *tasks are identified*:

1. Ac:
   a) acquire user cues from a camera;
   b) classify user's emotions;
   c) acquire emotions from a text.
2. Ex:
   a) generate affective text messages.
3. Ge:
   a) decay basic emotions exponentially based on personality;
   b) activate basic emotions based on personality;
   c) express basic emotions based on personality;
   d) appraise the affective state from the board of self;
   e) appraise the affective state from the board of others;
   f) decay emotions linearly;
   g) calculate appropriateness.
4. Ef:
   a) create social conclusions.
5. Rational tasks:
   a) make a move;
   b) regulate an emotion (intervene).

Tasks are then *grouped into the roles* by using guidelines above; there are no purely rational roles in this case.

Mixed roles:

- Social player: 2. a + 3. a + 3. b + 3. c + 3. d + 4. a + + 5. a;
- Regulator: 3.e + 3.f + 3.g + 5.b.

Affective roles:

- Sentiment analyser: 1.c;
- User state getter: 1.a + 1.b.

Then, based on the roles, *agents are defined*. User agent was made to represent a user due to technical limitations (embodiment) – if all agents tried to access a camera, it would slow down the system. Note that because of using this methodology in agents there are no conflicts in emotion dynamics. The set of agents is as follows:

- player (roles: a social player and a text reader);
- tutor (roles: a regulator and a text reader);
- user agent (role: a user state getter).

Interaction design (*both affective and rational*) consists of several interactions; in Fig. 3 emotion contagion (one of the subconscious mechanisms) and direct emotion expression (the conscious mechanism) are shown. There is also interaction between players and a user agent; a tutor and a user agent. *Ontology design* in this case is simple; it contains basic emotions and words "very" and "little", which are then used depending on the agent's internal state.

As mentioned before, the paper mostly focuses on the macro level; however, the micro level was also designed. In the micro level, *ECM* for students was designed that involved the OCEAN model personality impact on the activation, decay and expression functions, as well as affective state evaluation. *In architecture design*, to implement appraisal and social appraisal, multi-layered agent architectures for students, and deliberative agent architectures for a tutor and a user agent were used.

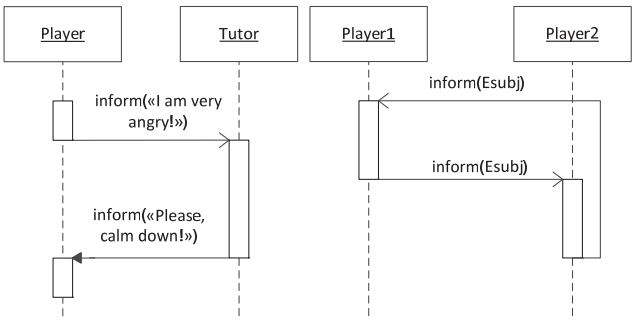

Fig. 3. On the left – interactions between a Player and a Tutor Agent; on the right – interaction between two Player Agents.

In the *deployment step*, it was important to decide on parameters for student personalities, which were chosen as an average based on psychology research; to implement various patterns, there were two types of social links: cooperative and competitive.

VI. Conclusion

The authors have found that the main challenges of design of affective MASs are the integration of agents with various affective capacities in one system, as well as affective interaction design. The paper offers high-level guidelines to tackle these issues. The approach is to separately design the rational tasks and corresponding roles from the affective tasks and the affective roles. Such an approach allows using a traditional AOSE approach to specify how the functional requirements will be met and only then analysing how the corresponding design will be affected by the affective tasks meeting the suprafunctional requirements. The guidelines also specify how each type of affective behaviour can be designed. Various AOSE methodologies can be used together with the proposed guidelines.

There is some research that can be conducted based on the presented study. One of the interesting questions is what kind of affective processes does the system need? Although rich emotional models allow the system to be believable and adaptive, it has never been considered that all systems need full affective capacity. While the paper answers how to use various affective abilities in the design phase that have been detected previously, it does not elaborate it on the requirements level. Similarly, the paper does not elaborate on how to determine whether all affective interaction mechanisms are needed as it is out of the scope of the paper.

References

[1] B. Henderson-Sellers and P. Giorgini, Eds., *Agent-Oriented Methodologies*, Hershey: Idea Group Publishing, 2005. https://doi.org/10.4018/9781591405818.ch001

[2] E. Lavendelis, "Open multi-agent architecture and methodology for intelligent tutoring system development," Summary of Doctoral Thesis, Riga: Riga Technical University, p. 49, 2009.

[3] M. Cossentino, V. Hilaire, A. Molesini and V. Seidita, Eds., *Handbook on Agent-Oriented Design Processes.* Berlin: Springer-Verlag Berlin Heidelberg, p. 577, 2014.
[4] Y. Demazeau, K. S. Desker, J. Bajo Perez and F. de la Prieta, Eds., "Advances in Practical Applications of Agents, Multi-Agent Systems, and Sustainability: The PAAMS Collection," *Proceedings of 13th International Conference, PAAMS 2015*, Salamanca, Spain, June 3–4, p. 323, 2015. https://doi.org/10.1007/978-3-319-18944-4
[5] H. K. Dam and M. Winikoff, "Towards a next-generation AOSE methodology," *Science of Computer Programming*, vol. 78, no. 6, pp. 684–694, 2013. https://doi.org/10.1016/j.scico.2011.12.005
[6] F. Bergenti, M. P. Gleizes and F. Zambonelli, Eds., *Methodologies and Software Engineering for Agent Systems. The Agent-Oriented Software Engineering Handbook*, London: Kluwer Academic Publishers, p. 505, 2004.
[7] S. A. DeLoach, "The MaSE Methodology," *Multiagent Systems, Artificial Societies, and Simulated Organizations*, vol. 11, pp. 107–125, 2004. https://doi.org/10.1007/1-4020-8058-1_8
[8] L. Padgham and M. Winikoff, *Developing Intelligent Agent Systems: A Practical Guide*. John Wiley & Sons, p. 240, 2004.
[9] E. Lavendelis, "Extending the MASITS Methodology for General Purpose Agent Oriented Software Engineering," *Proceedings of the 7th International Conference on Agents and Artificial Intelligence (ICAART),* January 10–12, Lisbon, Portugal, 2015, pp. 157–165. https://doi.org/10.5220/0005202201570165
[10] J. J. Odell. H. Van Dyke Parunak and B. Bauer , "Representing Agent Interaction Protocols in UML," *Lecture Notes in Computer Science*, pp. 121–140, 2002. https://doi.org/10.1007/3-540-44564-1_8
[11] "FIPA Ontology Service Specification," [Online]. Available: http://www.fipa.org/specs/fipa00086/
[12] K. Kravari and N. Bassiliades, "A Survey of Agent Platforms," *Journal of Artificial Societies and Social Simulation,* vol. 18, no. 1, p. 11, 2015 [Online]. Available: http://jasss.soc.surrey.ac.uk/18/1/11.html
[13] M. Wooldridge, N. R. Jennings and D. Kinny, "The Gaia Methodology for Agent-Oriented Analysis and Design," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no 3, pp. 285–312, 2000.
[14] P.M. Costa, T. Galvao, J. F. Cunha, and J. Pitt, "How to Support the Design and Development of Interactive Pervasive Environments," *Proceedings in 2015 8th International Conference on Human System Interaction, HSI 2015,* 2015.
[15] M. F. A. Carvalhaes, O. C. Da S. Neto, J. O. Ferreira, A. F. Da Rocha and T. M. G. De A. Barbosa, "Including Affectivity Requirements in Embedded Systems," *Proceedings of the 7th Annual IEEE International Systems Conference SysCon 2013*, pp. 229–34, 2013. https://doi.org/10.1109/syscon.2013.6549887
[16] L. Axelrod and K. Hone, "Smoke and Mirrors: Gathering User Requirements for Emerging Affective Systems," *Proceedings of the International Conference on Information Technology Interfaces ITI 2004,* June 7–10, Cavtat, Croatia, pp. 323–328, 2004.
[17] C. Reynolds and R.W. Picard, "Designing for Affective Interactions," *Proceedings of the 9th International Conference on Human-Computer Interaction*, 2005.
[18] R. GhasemAghaei, A. Arya and R. Biddle, "Design Practices for Multimodal Affective Mathematical Learning," *Proceedings of the 20th International Symposium on Computer Science and Software Engineering (CSSE) 2015,* 2015. https://doi.org/10.1109/csicsse.2015.7369246
[19] Y. S. Sefidgar, K. E. MacLean, S. Yohanan, H. F. M. van der Loos, E. A. Croft and E.J. Garland, "Design and Evaluation of a Touch-Centered Calming Interaction with a Social Robot," *IEEE Transactions on Affective Computing*, vol. 7, no. 2, pp. 108–121, 2015. https://doi.org/10.1109/taffc.2015.2457893
[20] E. Hudlicka, "Guidelines for Designing Computational Models of Emotions," *International Journal of Synthetic Emotions*, vol. 2, no. 1, pp. 26–79, 2011. https://doi.org/10.4018/jse.2011010103
[21] C. Smith, N. Crook, J. Boye, D. Charlton, S. Dobnik, D. Pizzi, M. Cavazza, S. Pulman, R. S. De La Camara and M. Turunen, "Interaction Strategies for an Affective Conversational Agent," *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 301–314, 2010. https://doi.org/10.1007/978-3-642-15892-6_31
[22] R. Calvo, S. D'Mello, J. Gratch, A. Kappas, R. Reisenzein, "A Short History of Psychological Perspectives on Emotion," *The Oxford Handbook of Affective Computing*, pp. 1–20, 2015. https://doi.org/10.1093/oxfordhb/9780199942237.013.014
[23] P. Ekman, "Are There Basic Emotions?" *Psychological Review*, vol. 99, no. 3, pp. 550–553, 1992.
[24] J. A. Russell and A. Mehrabian "Evidence for a Three-Factor Theory of Emotions," *Journal of Research in Personality*, vol. 11, no. 3, pp. 273–294, 1977.
[25] A. Ortony, G. L. Clore, and A. Collins, "Introduction," *The Cognitive Structure of Emotions*, Cambridge University Press, 1988. https://doi.org/10.1017/cbo9780511571299.002
[26] M. Pudane, "Affective Multi-Agent System for Simulating Mechanisms of Social Effects of Emotions," *Proceedings of the 7th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW),* 2017.
[27] S. Petrovica and M. Pudane, "Emotion Modeling for Simulation of Affective Student-Tutor Interaction: Personality Matching," *Int. J. Educ. Inf. Technol.*, vol. 10, pp. 159–167, 2016.
[28] Affectiva, [Online]. Available: https://www.affectiva.com/
[29] R. Z. Cabada, M. L. B. Estrada, V. J. A. Beltr'n, C. A. Reyes García, and Y.H. Pérez, "Fermat: Merging Affective Tutoring Systems with Learning Social Networks," *2012 IEEE 12th International Conference on Advanced Learning Technologies*, pp. 337–339, 2012. https://doi.org/10.1109/icalt.2012.140
[30] P. Gebhard, "ALMA – A Layered Model of Affect", *AAMAS '05: Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systemsent Systems*, pp. 29–36, 2005.
[31] C. Becker-Asano, "WASABI : Affect Simulation for Agents with Believable Interactivity," Doctoral Thesis, Universitat Bielefeld, 2008.
[32] M. Tavakoli, M. Palhang and M. Kazemifard, M, "Three levels of information processing: Improvement using personality," *2014 Iranian Conference on Intelligent Systems, ICIS 2014*, 2014. https://doi.org/10.1109/iraniancis.2014.6802602
[33] S. Korecko, B. Sobota and P. Curilla, "Emotional agents as non-playable characters in games: Experience with Jadex and JBdiEmo," *Proceedings of the 15th IEEE International Symposium on Computational Intelligence and Informatics (CINTI) 2014,* pp. 471–476, 2014. https://doi.org/10.1109/cinti.2014.7028721
[34] J. Hastings, W. Ceusters, B. Smith, and K. Mulligan, "Dispositions and processes in the emotion ontology," *CEUR Workshop Proceedings*, vol. 833, pp. 71–78, 2011.

Māra Pudāne is a Research Assistant and Doctoral student of the study programme "Computer Systems" at Riga Technical University. She obtained the Master's degree in Computer Systems from Riga Technical University, Latvia, in 2013. During her Master studies, she started to work as a Research Assistant at the Department of Artificial Intelligence and System Engineering (Riga Technical University). The topic of her Doctoral Thesis is related to human group behaviour imitation with a focus on affective factors. Ms. Pudane's research interests include human modelling, multi-agent systems and affective computing.
E-mail: mara.pudane@rtu.lv

Egons Lavendelis is an Associate Professor and Researcher in the area of Artificial Intelligence. Egons defended his Doctoral Thesis in the area of agent-oriented software engineering proposing the MASITS methodology at Riga Technical University (RTU) in 2009. After the defence, he has been working as a Senior Researcher and Assistant Professor at the Department of Artificial Intelligence and Systems Engineering. Egons's research interests include multi-agent systems, agent-oriented software engineering, communication among autonomous entities, including semantics, different applications of agent paradigm and various artificial intelligence techniques; lately his attention has been devoted to the use of multi-agent systems for integration of autonomous robots into collaborative multi-robot systems. At present, he is delivering study courses related to artificial intelligence and databases at RTU. He is the author of more than 30 articles published in various international conference proceedings, scientific journals, and book chapters.
E-mail: egons.lavendelis@rtu.lv