Selecting and Expressing Communicative Functions in a SAIBA-Compliant Agent Framework

Angelo Cafaro ✉[2], Merijn Bruijnes ✉[1], Jelte van Waterschoot[1], Catherine Pelachaud[2], Mariët Theune[1], and Dirk Heylen[1]

[1] Human Media Interaction, University of Twente, The Netherlands
[m.bruijnes, d.k.j.heylen, m.theune, j.b.vanwaterschoot]@utwente.nl
[2] CNRS-ISIR, Pierre and Marie Curie University, France
[cafaro, pelachaud]@isir.upmc.fr

Abstract. In SAIBA-compliant agent systems, the Function Markup Language (FML) is used to describe the agent's communicative functions that are transformed into utterances accompanied with appropriate non-verbal behaviours. In the context of the ARIA Framework, we propose a template-based approach, grounded in the DIT++ taxonomy, as an interface between the dialogue manager (DM) and the non-verbal behaviour generation (NVBG) components of this framework. Our approach enhances our current FML-APML implementation of FML with the capability of receiving on-the-fly generated natural language and socio-emotional parameters (e.g. emotional stance) for transforming the agent's intents in believable verbal and non-verbal behaviours in an adaptive manner.

Keywords: Dialogue management, communicative function, FML, multimodal behaviour, SAIBA

1 Introduction

Generating natural multimodal behaviour for an embodied conversational agent requires producing utterances accompanied with appropriate non-verbal behaviours and the capability to 'colour' these behaviours to adapt to the social situation [2, 15]. In a SAIBA system [7], an *Intent Planner* produces intents composed of communicative functions (and the topic) that are translated into expressive multimodal behaviours by a *Behaviour Planner*. This is typically the joint task of a Dialogue Manager (DM) on the one hand (intent planner) and a non-verbal behaviour generation (NVBG) system on the other (behaviour planner). The challenge is to dynamically create behaviour that is believable and fits the social situation.

An author can manually craft a dialogue scenario to control the display of a believable agent's social behaviour. However, this is a rigid approach that requires authoring of each utterance and, even when several variants for each utterance exist, it is likely there will be not enough variability to accommodate all social situations. In a more dynamic approach, it is difficult to retain control of the results, which might lead to unbelievable generated content. It is hard to automatically generate (non)verbal behaviour

J. Beskow et al. (Eds.): IVA 2017, LNAI 10498, pp. 73-82, 2017.
DOI 10.1007/978-3-319-67401-8_8

supporting, for instance, emphasis on words. For easy authoring, while escaping rigidity of pre-scripted files, we propose re-usable scripts, templates, that offer flexibility in the way their content is delivered [17].

In this paper, we propose a SAIBA-compliant interface between a DM and an NVBG system that allows the DM to dynamically instantiate the communicative functions that are sent to the NVBG. We follow a template-based approach that builds on the DIT++ taxonomy of communicative functions [3]. DIT++ supports dynamic variability of produced content, both verbal and non-verbal behaviour, while ensuring a certain degree of control over the resulting behaviours. The main challenge in this approach is defining templates that have appropriate placeholders where the system can enter or modify variables to create appropriate behaviour. Additional challenges are the automatic selection of these templates and setting the value of the variables.

The contribution of this paper is threefold: (1) We propose an enhancement to FML through the definition of FML Templates that serve as an interface between a DM and a SAIBA-compliant NVBG platform. (2) We describe the mechanism adopted by the DM to fill the placeholders provided in the FML templates. (3) We provide insights into the benefits of using a template-based FML representation within the SAIBA framework.

2 Related Work

The work presented in this paper is part of the *Artificial Retrieval of Information Assistants – Virtual Agents with Linguistic Understanding, Social skills, and Personalised Aspects* (ARIA-VALUSPA) project[3] in which we create agents that are capable of holding multimodal social interactions in challenging and unexpected situations. The demo scenario is Alice in Wonderland where the agent portrays Alice. In this paper, we describe the interface between the DM and the NVBG system of the ARIA-VALUSPA framework depicted in Figure 1 and described in Section 3.1. We have grounded our work in two standards, respectively, for describing communicative functions (DIT++) and for representing them (FML-APML).

The **Dynamic Interpretation Theory** (DIT++) taxonomy, an ISO standard [3], is a comprehensive, application-independent system for classification and analysis of dialogue with information about the communicative acts that are performed by dialogue segments (a turn can be a dialogue segment). DIT++ has been used before in the design of a DM module [6]. The **FML-APML** is an evolution of the Affective Presentation Markup Language (APML) [13] and is used by the NVBG system component in the ARIA framework described in Section 3.1. The original FML-APML tags encoded the communicative intentions of an agent following the categorization of Poggi [16]. Contrary to the representation proposed by Cafaro et al. [4], the set of tags in the FML-APML supports features regarding the timing and importance of communicative functions. The timing is specified with attributes inspired by the BML recommendations [7] and makes possible absolute or relative timings of functions with symbolic labels for referencing. Emotional states can be described and these tags also give the possibility to specify an *intensity* (from 0 to 1).

[3] `https://aria-agent.eu`

The dialogue management component in our work extends Flipper [11], a DM based on the principles from the TrindiKit DM as an information state based DM [8]. Other DM systems have been proposed lately. Rich and Sidner proposed DISCO [17], a task-based DM based on collaborative discourse theory. In addition to offering content-wise placeholders such as DISCO, our approach also supports a parametric instantiation of accompanying communicative functions (e.g. emotion and level of emphasis). Morbini et al. have created FLoReS [14], a DM that facilitates the creation of structured dialogues with the use of domain experts. We use a similar approach with forward-looking goals in the form of communicative functions, though our approach differs from FLoRes in the agent's intent generation, where we update our information state in real-time with topics of interest and emotional state of the user and insert these in behavioural templates. OpenDial is a toolkit for developing spoken dialogue systems created by Lison [10]. His main contribution was adding the possibility to learn probabilities for the responsive behaviour of the agent, even with small amounts of data, and still make it easy to author the dialogue models. The authoring of dialogues is similar to that of dialogues created with our DM Component with high-level templates, with placeholder and mapping between utterances and intents. The Virtual Human Toolkit uses question answering algorithms to select the agent's response. However, they do not utilize an information state, or an agent mental state, to alter the responses of the agent [9]. Finally, Mairesse and Walker developed PERSONAGE [12], a parametrizable NLG tool that produced text outputs varying along the extraversion personality dimension of the Big 5 [5]. In PERSONAGE a broader set of parameters is employed compared to our system, however the authors' focus is language generation whereas we propose a richer output that includes instances of speech acts (i.e. language) but is also supported by a variety of socio-emotional communicative functions later transformed into multimodal behaviour.

3 An FML-Template based Dialogue Manager

3.1 The ARIA Framework

The ARIA framework has an architecture composed of three major blocks of modules: Input, Agent Core and Output, as shown in Figure 1. Each block itself consists of a number of modules. The Input block is mainly responsible for collecting and processing audio-visual data about the user. The Social Signal Interpretation framework (SSI) [18] in the input block gathers multimodal audio-visual user data and interprets, for instance, the user's emotional state in terms of valence and arousal, and the text uttered by the user. This data is then fed to the Agent Core which analyses it to decide on the agent's response behaviour. The main component of the Agent Core is the Dialogue Manager (DM), which is an evolution of Flipper [11]. Based on the input from SSI, the Core block produces FML-APML scripts that serve as input for the Output generation block. In the Output block, the NVBG component is responsible for rendering the agent, displaying animated behaviour and playing synthesized speech using the CereVoice Engine text-to-speech (TTS) tool developed by CereProc [1]. The three blocks use ActiveMQ as a message broker for communication.

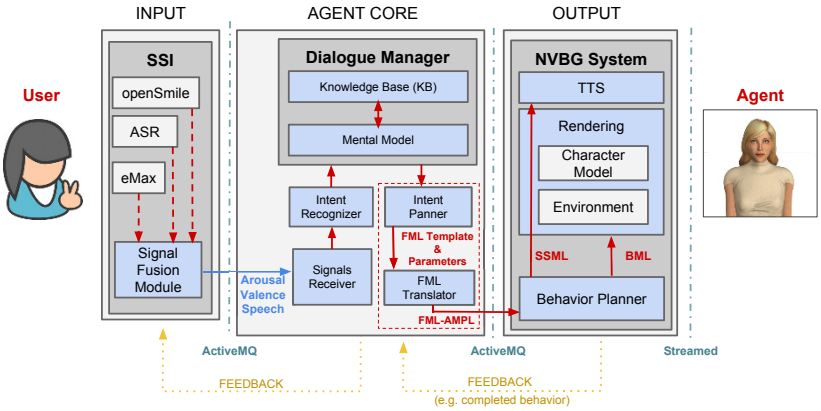

Fig. 1. Overview of the ARIA Framework architecture.

This paper focuses on the interface between the Agent Core and the Output generation block, also highlighted in Figure 1 by the dashed red square. More specifically, we describe the working mechanism of our DM (Section 3.2) and the mechanism that allows it to interface with the NVBG component (Sections 3.3 and 3.4)[4].

3.2 FML-Templates

The DM decides when the agent needs to communicate with the user and what to say. Expressing this intent is done via the FML Translator, which communicates with the NVBG component in order to generate, in real-time, the appropriate verbal and non-verbal behaviour. The DM provides the FML Translator two types of information: (1) an FML Template, and (2) a set of parameters that depend on the selected template. These parameters are retrieved from the information state to add variability in the behaviour specified in the FML template. In this section we describe the set of FML Templates that we have created, the available parameters in these templates, the process to choose a template and the values for its parameters and the final transformation to FML-APML.

FML Templates are based on the DIT++ taxonomy and are categorized accordingly. The DIT++ taxonomy describes dialogue segments in terms of communicative functions. When the agent receives user input the communicative function of this input can be used to formulate an appropriate response. A dialogue segment can have multiple functions. We split up such segments into smaller units so that only one communicative function remains per unit. For example, assuming that the user asks a question (e.g. a DIT++ *set question*) as interpreted by the Intent Recognizer which is part of the Agent Core (see Fig. 1), an agent's response can indicate that it understands there was a question (i.e. *positive feedback* function) and, at the same time, can be a reply to the question (computed by the Intent Planner as an *inform* function as a response). For each of the

[4] The code is provided at: `https://github.com/ARIA-VALUSPA/ARIA-System`. Additionally, an example scenario is provided at: `https://github.com/ARIA-VALUSPA/ARIA-System/wiki/Documentation`.

relevant communicative functions contained within DIT++, we created an FML Template with a subset of parameters (described in the following section). An overview of the DIT++ communicative functions in our FML Templates is shown in Table 1.

In addition to the existing DIT++ communicative functions *answer*, *agreement*, and *disagreement* (subfunctions of *inform*), we need additional *inform* functions in order to fine-tune the way the agent provides information. DIT++ supports adding more specific communicative functions [3], so we introduce two more inform subfunctions, *elaborate* and *explain*, which are used to give more examples and an explanation of a topic.

Table 1. An overview of our FML-Templates categorized according to DIT++ taxonomy.

Class	**Goal**	**Sub-classes**
Information Transfer	Obtain or provide information	Question: set | choice | prop | check Inform: agreement | disagreement | answer | elaborate | explain
Feedback	Provide or elicit information about the processing of the previous utterance(s)	Auto: positive | negative Allo: positive | negative Elicitation
Interaction	Structure the dialogue (e.g. turn or topic management)	Contact: check | indication Time: stalling | pausing Turn: take | accept | grab | keep | assign | release Topic: introduction | preclosing | announceShift
Social Obligations	Social policies during the dialogue	Salutation: initial | return Introduction: initial | return Gratitude: initial | return Apology: initial | return Valediction: initial | return

FML Template Parameters. The DM can modify parameters in FML Templates to 'colour' the agent's behaviour. The parameters are in the form of XML elements and their attributes as shown in Table 2. The **Element** column indicates the name of an element as it appears in the FML Template. The three last elements enhance FML-APML with additional constructs supporting placeholders for adding variability as described in

Table 2. The parameters of our FML Templates are elements and changeable attribute values.

Element	**Attribute**
emotion	type, intensity, importance
emphasis	level, importance
certainty	type, intensity, importance
voice	type
var	type
alternative	type, name
alt-option	ref

this section. The **Attribute** column indicates an element's attribute that has a selectable or changeable value (i.e. by the DM). The attribute *type* of a **<var>** element can be: *sentence*, *topic*, *user*, or *agent*. The *type* for **<certainty>** can be *none*, *uncertain*, or *certain*. The **<alternative>** elements have a *type* attribute that can be: *static*, *dynamic*, or *selectable*. The *name* attribute is a string used to link multiple selectable alternative tags by name. The **<alt-option>** elements can be children of selectable alternatives and their *ref* attribute is used for choosing a specific one. When multiple selectable alternatives have the same name (i.e. they are linked), the given alt-option is selected in all linked selectable alternatives whereas the other alt-options are discarded (see the example in the next section for more details). The attributes *intensity*, *importance*, and *level* are float values ([0..1]).

Standard FML-APML tags. The **<emotion>** tag has a *type*, *intensity* and *importance* attribute. This tag sets the emotion the agent should express and it can be combined with other communicative functions. The **<emphasis>** tag, if it is present in a template, emphasizes verbally and non-verbally a defined part of the agent's speech. The **<certainty>** tag allows the DM to specify whether a communicative function should be expressed (via non-verbal behaviours) with certitude or incertitude. The **<voice>** tag has originally been defined within the CereVoice Engine to synthesize speech with different emotional stances. It is now included in FML-APML and its attribute, named type, can have four possible values supported by the CereVoice Engine (angry, happy, calm, sad).

Additional tags. Our FML Templates contain additional tags augmenting FML-APML. We defined these additional tags on top of the standard FML-APML ones to overcome the limit of using pre-scripted FML-APML instances and to support more variability in dialogue and behaviour generation phases. We created tags that are placeholders for words and full sentences, and tags that are constructs for adding variability to the predefined result (i.e. the FML script that is sent to the NVBG system).

A **<var>** is a placeholder for constituents. For instance a sentence, a topic, or the name of the user or agent. One example use is self-referencing of the agent: "Hello, I am **<var type="agent">**".

An **<alternative>** is a placeholder that supports alternative texts or FML-c blocks of elements that can be selected. Three alternative types exist: *static*, *selectable* and *dynamic*. A **static** alternative contains a list of child tags that include FML content. When present, one of the alt-option children is randomly selected with a uniform distribution. A static alternative allows the DM to fully delegate the generation of the FML to the FML Translator. However, changes or additions to this construct need to be scripted *before runtime*. An example of a static alternative tag is shown in listing 1.1.

```
<alternative id="alt1" type="static">
 <alt-option>For <tm id="tm1"/>instance:</alt-option>
 <alt-option>For <tm id="tm1"/>example:</alt-option>
</alternative>
```

Listing 1.1. Example of a static alternative.

With a **selectable** alternative it is possible to conditionally select one of the alt-options that are available via the *ref* attribute, as opposed to randomly selecting one of them. The Agent Core has full control over the resulting FML-APML sent to the NVBG by choosing which alternative should be produced. Multiple chunks of selectable alternatives can be linked together if they have the same *name* attribute. The result is that the selected alt-option (via *ref*) is also selected for the other linked selectable alternatives. In the following example, giving Bob as input value for the user name (i.e. var) the "named" alt-option of the *positive-feedback* selectable alternative yields to "Yes Bob" when chosen. The emphasis tag belonging to the named alt-option in the second alternative (i.e. alt2) is also selected as it belongs to a linked alternative, see listing 1.2.
A **dynamic** alternative receives as input a list of semicolon separated items (e.g. words) of which one is randomly selected with a uniform distribution probability. This allows the DM to change the possible variations *at runtime*, but it has the disadvantage of requiring the DM to provide actual text content. For example, the dynamic alternative tag shown below could take from the DM a list of items such as: *"bike; car; foot"* (see listing 1.3).

```
<speech id="s1">
<alternative id="alt1" name="positive-feedback" type="selectable">
 <alt-option ref="named">Yes <tm id="tm0"/><var id="var1" type="user"/><tm id="tm1
      "/></alt-option>
 <alt-option ref="no-named">Yes</alt-option>
</alternative>
</speech>
<alternative id="alt2" name="positive-feedback" type="selectable">
 <alt-option ref="named"><emphasis id="emp1" start="s1:tm0" level="strong" end="
      s1:tm1" importance="1"/></alt-option>
 <alt-option ref="no-named"></alt-option>
</alternative>
```

Listing 1.2. Example of a selectable alternative.

```
<tm id="tm0"/>
 <alternative id="alt1" type="dynamic"/>.
<tm id="tm1"/>
```

Listing 1.3. Example of a dynamic-alternative.

Finally, **<var>** and **<alternative>** tags can be **nested**. It is possible to create nested structures by including within any type of alternative: **<var>** elements or other **<alternative>** types (n.b. within one level of recursion though).

3.3 Dialogue Management as FML Template Selection

Selecting and modifying FML Templates is described as a pipeline. After a new utterance has been detected by SSI, the Intent Recognizer computes the user's communicative functions. More specifically, an NLP component extracts keywords (based on nouns/verbs/adjectives) to determine the topic of the user's utterance and the communicative functions according to DIT++. The result of this computation is also stored in the agent's mental model. Next, the Intent Planner in the Agent Core has internal precondition rules that are activated when matching with the user's communicative function and topic of interest. Each FML Template is coupled to a unique set of preconditions and once those are activated, the corresponding FML Template is chosen.

Depending on the Template, the Intent Planner needs to select the appropriate values for the available parameters. Those are retrieved from the agent's mental state. The agent's utterances are currently stored in an internal database and are constrained to the Alice in Wonderland scenario. Additionally, the agent's utterance can be augmented with an emotional expression (e.g. a frown or a raise in pitch when angry). Our DM supports both a static way of defining the agent's emotion (e.g. always happy, sad) or a more dynamic way (e.g. mirroring the user's emotion, or taking the output of computational models of emotions).

Once all parameters have been retrieved, the FML Translator takes the FML Template and the required parameters as input and generates a full FML-APML script that is compliant with the SAIBA behaviour planner within the NVBG system.

3.4 FML Translator

The job of the FML Translator is to transform a given FML template and its input parameters into an FML-APML script that is processable by the NVBG system for generating synthesized speech and accompanying non-verbal behaviour. The FML Translator algorithm takes the following steps to accomplish this transformation task:

1. Find *selectable* **<alternative>** elements and replace them with the selected alternative-item's content.
2. Find *dynamic* **<alternative>** elements, randomly choose an alternative from the list of items given in input and replace it.
3. Find *static* **<alternative>** elements, randomly choose an alternative's content and discard the others in the final FML-APML script.
4. Find **<var>** elements and replace those according to the given DM input.
5. Find **<voice>** tags (for CereVoice) and replace the emotion attribute OR remove the voice brackets if input is not given.
6. Find and replace values for FML attributes (e.g. emotion *type* and *intensity*).

Behaviour Generation. The NVBG system is a SAIBA-compliant platform that expects to receive communicative functions from an intent planner represented in FML. The FML-APML implementation of FML is currently used. FML-APML input is transformed to BML (i.e. behaviour) according to a *Multimodal Behaviour Lexicon* and probabilistic rules. The *lexicon* can be seen as a dictionary in which an entry is a communicative function (described with category and type). For each entry (i.e. function), a set of behaviours (involving facial expressions, gaze, gestures, etc...) to accomplish the function is proposed along with several alternatives that are named *behaviour sets*. Each behaviour set comes with a probability determining the likelihood of being chosen among the others in the entry. A basic lexicon can have a discrete uniform distribution associated to each behaviour set, for example, each alternative has an equal probability of being chosen with respect to the other alternatives in the same entry. As a result, any given set of communicative functions represented in FML-APML can be accomplished in different ways.

4 Conclusions and Future Work

We have created an interface that offers variability content-wise, but also at a functional level thanks to the possibility of choosing, for example, the emotional expression and emphasis. In general, basing the templates on a taxonomy of communicative functions provides a solid background for the DM to work with. The templates do not only deal with generated natural language, but also include a standard representation of communicative functions which makes transforming the given input into multimodal generated verbal and non-verbal behaviour a simpler task for the NVBG system. In our ARIA Framework, the agent combines basic NLP, social signal detection, and communicative functions planning to optimally structure the dialogue with the user. The system is modular and can be extended (e.g. with emotion and natural language generation engines) to achieve more flexibility and variability.

We have proposed a mixed approach of control over audio-visual results of generated multimodal behaviour while still leaving room for variability. It should be remarked that the more dynamically content has to be generated (e.g. using dynamic alternatives) the more intelligent the decision-making of the DM needs to be. For instance, automatically filling a dynamic alternative list with appropriate verbal content requires natural language understanding: the agent needs to respond appropriately to the content of the user's utterance. Static alternatives offer the advantage of yielding more controlled results by reducing the burden of decision-making in the DM and delegating it to the transformation phase (i.e. FML Translator). However, variability can only be obtained with costly off-line authoring. Finally, selectable alternatives represent a compromise between the two. It allows an author to prepare what can be said while offering the DM the capability to select appropriate behaviour.

Authoring of a dialogue scenario is one of the main efforts when developing an agent in most frameworks. By utilizing FML Templates, the author can reuse dialogue features that occur often. In addition, as a dialogue scenario grows over time, the pool from which to pick a template to reuse becomes larger.

Some limitations need to be addressed in future work. First, a sentence provided as input clause is not divided in smaller segments of information. The DM should be able to point to parts of a sentence with more accuracy. This would allow the agent to refer to specific information from a segment of a sentence it has said. This is relevant, for example, when the agent is interrupted and needs to determine whether the information contained in a segment was understood. We plan to overcome this issue by adding a step to the transformation process that takes into account the presence of special markers within a sentence indicating specific portions to emphasize, and make more precise the generated emphasis behaviour. Finally, authoring of templates, attributes, and values can become complex with large dialogue scenarios. A GUI editor would benefit our approach and the other SAIBA-compliant agent systems, and would make authoring considerably easier. Developments towards such an editor are under way.

Acknowledgements

This work is supported by the European project H2020 ARIA-VALUSPA. We are grateful to Alexandru Ghitulescu for his help in developing the FML Translator.

References

1. Aylett, M., Pidcock, C.: The CereVoice Characterful Speech Synthesiser SDK. In: Pelachaud, C., Martin, J.C., André, E., Chollet, G., Karpouzis, K., Pelé, D. (eds.) Intelligent Virtual Agents, LNCS, vol. 4722, pp. 413–414. Springer (2007)
2. Bruijnes, M.: Believable suspect agents: response and interpersonal style selection for an artificial suspect. Ph.D. thesis, University of Twente (2016), sIKS dissertation no. 2016-39
3. Bunt, H., Alexandersson, J., Choe, J.W., Fang, A.C., Hasida, K., Petukhova, V., Popescu-Belis, A., Traum, D.R.: Iso 24617-2: A semantically-based standard for dialogue annotation. In: LREC. pp. 430–437 (2012)
4. Cafaro, A., Vilhjálmsson, H., Bickmore, T., Heylen, D., Pelachaud, C.: Representing Communicative Functions in SAIBA with a Unified Function Markup Language. In: Bickmore, T., Marsella, S., Sidner, C. (eds.) Intelligent Virtual Agents, Lecture Notes in Computer Science, vol. 8637, pp. 81–94. Springer International Publishing (2014)
5. Goldberg, L.R.: An alternative "description of personality": the big-five factor structure. Journal of Personality and Social Psychology 59(6), 1216–1229 (1990)
6. Keizer, S., Bunt, H., Petukhova, V.: Multidimensional dialogue management. In: van den Bosch, A., Bouma, G. (eds.) Interactive Multi-modal Question-Answering, pp. 57–86. Springer (2011)
7. Kopp, S., Krenn, B., Marsella, S., Marshall, A.N., Pelachaud, C., Pirker, H., Thórisson, K.R., Vilhjálmsson, H.H.: Towards a common framework for multimodal generation: The behavior markup language. In: Proceedings of the 6th international conference on Intelligent Virtual Agents. pp. 205–217. IVA'06, Springer-Verlag, Berlin, Heidelberg (2006)
8. Larsson, S., Traum, D.R.: Information state and dialogue management in the TRINDI Dialogue Move Engine Toolkit. Natural Language Engineering 6(3&4), 323–340 (2000)
9. Leuski, A., Traum, D.: NPCEditor: Creating Virtual Human Dialogue Using Information Retrieval Techniques. AI Magazine 32(2), 42–56 (Jul 2011)
10. Lison, P.: Structured probabilistic modelling for dialogue management. Ph.D. thesis, University of Oslo (2013)
11. ter Maat, M., Heylen, D.: Flipper: An Information State Component for Spoken Dialogue Systems. In: International Workshop on Intelligent Virtual Agents. pp. 470–472 (2011)
12. Mairesse, F., Walker, M.: PERSONAGE: Personality Generation for Dialogue. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics. vol. 45, pp. 496–503. Association for Computational Linguistics (2007)
13. Mancini, M., Pelachaud, C.: The FML-APML language. In: Why Conversational Agents do what they do. Workshop on Functional Representations for Generating Conversational Agents Behavior at AAMAS (2008)
14. Morbini, F., DeVault, D., Sagae, K., Gerten, J., Nazarian, A., Traum, D.: FLoReS: A Forward Looking, Reward Seeking, Dialogue Manager. In: Natural Interaction with Robots, Knowbots and Smartphones, pp. 313–325. Springer (2014)
15. Ochs, M., Sabouret, N., Corruble, V.: Simulation of the dynamics of nonplayer characters' emotions and social relations in games. IEEE Transactions on Computational Intelligence and AI in Games 1(4), 281–297 (2009)
16. Poggi, I.: Mind, hands, face and body: A goal and belief view of multimodal communication. Weidler Buchverlag Berlin (2007)
17. Rich, C., Sidner, C.L.: Using Collaborative Discourse Theory to Partially Automate Dialogue Tree Authoring. In: Proceedings of the 12th International Conference on IVAs. pp. 327–340. Springer-Verlag, Berlin, Heidelberg (2012)
18. Wagner, J., Lingenfelser, F., Baur, T., Damian, I., Kistler, F., André, E.: The social signal interpretation (SSI) framework: multimodal signal processing and recognition in real-time. In: Proceedings of the 21st ACM International Conference on Multimedia. pp. 831–834 (2013)