Towards a Common Framework for Multimodal Generation: The Behavior Markup Language

Stefan Kopp[1], Brigitte Krenn[2], Stacy Marsella[4], Andrew N. Marshall[4], Catherine Pelachaud[3,] Hannes Pirker[2], Kristinn R. Thórisson[5], Hannes Vilhjálmsson[4]

[1] Artificial .Intelligence Group, University of Bielefeld, Germany
skopp@techfak.uni-bielefeld.de
[2] Austrian Research Institute for AI (OFAI), Vienna, Austria
{brigitte, hannes}@ofai.at
[3] IUT de Montreuil, University de Paris 8, France
c.pelachaud@iut.univ-paris8.fr
[4]Information Sciences Institute, University of Southern California USA
{marsella, amarshal, hannes}@isi.edu
[5]CADIA, Dept. Of Computer Science, Reykjavik University, Iceland
thorisson@ru.is

Abstract. This paper describes an international effort to unify a multimodal behavior generation framework for Embodied Conversational Agents (ECAs). We propose a three stage model we call SAIBA where the stages represent intent planning, behavior planning and behavior realization. A Function Markup Language (FML), describing intent without referring to physical behavior, mediates between the first two stages and a Behavior Markup Language (BML) describing desired physical realization, mediates between the last two stages. In this paper we will focus on BML. The hope is that this abstraction and modularization will help ECA researchers pool their resources to build more sophisticated virtual humans.

1 Introduction

Human communicative behaviors span a broad set of skills, from natural language generation and production, to coverbal gesture, to eye gaze control and facial expression. People produce such multimodal behavior with ease in real-time in a broad range of circumstances. The simulation of such behaviors with computer-generated characters has, by now, a history of more than ten years [15][1]. A number of approaches have been presented in the field, geared toward specific aspects of generating multimodal behavior, e.g. facial expressions and gesture synthesis. All represent models of a production process in which certain knowledge structures are identified and transformed. Such knowledge structures include representations of communicative intent, lexicons that define available behaviors and their particular overt forms, and rules as to how communicative intent and affective state is mapped onto them.

At the AAMAS 2002 workshop "Embodied conversational agents - let's specify and evaluate them!" it became obvious that most researchers were building their own behavior and functional languages. While diversity is important, another "Gesticon" workshop in 2003 made it clear that a lot of similarities existed among the approaches. To avoid replication of work, as well as to allow for sharing modules, a push was initiated to develop a common specification. In April 2005, a group of researchers in the area of multimodal communication and computer animation came together at Reykjavik University to further the integration and development of multimodal generation skills for artificial humans [18]. Our goals were (1) to frame the problem of multimodal generation in a way that allows us to put it into computational models; (2) to define planning stages of multimodal generation and to identify the knowledge structures that mediate between them; (3) to render these stages and knowledge structures into a framework that lays down modules and interfaces, enabling people to better work together and to use each other's work, that has been directed to different aspects of multimodal behavior, with a minimal amount of custom work. In previous efforts we started by clarifying terminologies such as representation vs. markup vs. scripting languages [9].

In this paper we describe our latest results in this ongoing process. In Section 2, we begin by looking into four existing languages: BEAT, MURML, APML and RRL. Our goal is to bring together our experiences with these languages and to derive a powerful, unifying model of representations for multimodal generation. We present such a model, the SAIBA framework, in Section 3. Two important representation languages emerged as part of this framework. These languages are meant to be application independent, graphics model independent, and to present a clear-cut separation between information types (function versus behavior specification). We will go into one of those languages, the Behavior Markup Language (BML), in more detail in Section 4, and then conclude with remarks on the next steps.

2 Prior approaches

A number of researchers have construed representation languages for capturing the knowledge structures that were identified as involved in the generation of multimodal behavior. We start here by analyzing four broadly used languages, all being XML compliant. While there are certainly more languages being employed out there (e.g. MPML; [12]), the languages considered here provide a good overview of previous approaches, and allow us to compare the assumptions that underlie their generation models.

One principal commonality among these and related previous systems is the separation of content- and process-related processing. For example, the Ymir architecture used to implement the Gandalf humanoid clearly separated dialog planning and social interaction control [16][17]. The argument behind this was that what an agent chooses to say in a given situation is highly domain-specific, whereas the ability to deliver that content through social interaction is a broad re-usable skill. Consequently, verbal responses related to dialog topic (content) were generated by a separate process, based on the user's interpreted communicative act (the multimodal version of a speech act), using an abstract frame-based representation. The surface form, however, of this content and all necessary process-related responses (turntaking signals, gaze, head movements, gesture, paraverbals), was generated by a realtime, rule-based planner (called Action Scheduler) in incremental chunks of 200-1200 msec duration each, using a library of composite behaviors (called Behavior Lexicon). The whole process was driven by a pervasive representation of time and the high-level abstract communicative goals. The separation of content and process was kept in the architecture of the later Rea system [3]. There, a special generation module was dedicated to verbal and nonverbal behavior generation, taking an abstract representation of communicative intent and giving it surface form according to the rules of social face-to-face interaction. One consequence of these systems' emphasis on modularization has been that formal and re-usable representations that interface between separated stages moved into the focus of research on the automatic generation of multimodal behavior.

2.1 BEAT/Spark

The BEAT "text-to-embodied-speech" toolkit [2] specifically addressed this re-use issue by introducing a plug-in model for nonverbal behavior generators and an XML-based processing pipeline. The pipeline has clear stages that move representations from annotations of communicative intent to behavior suggestions and finally to scheduling and execution. Yet, the behavior generators have access to a variety of information about the text to be spoken, at different levels of abstraction, and therefore don't quite provide a clean interface to communicative intent. The Spark system modified BEAT to work within an avatar-based chat system, using the behavior generators to automatically animate the delivery of text messages between chatting participants[19][20]. The division between communicative intent and behavior was made very clear with Spark's definition of two separate XML tag sets. The text messages are first annotated automatically with tags, describing communicative intent or functions (function markup), and then the generators transform those tags into another set of tags (behavior markup), turning communicative functions into the behaviors that support them. The XML annotation is all done inline with the spoken text and while that makes temporal co-occurrence easy to process, it does not allow partially overlapping temporal spans.

2.2 MURML

In the MAX system (or, more generally, the *Articulatd Communicator Engine*) [6], a *Multimodal Utterance Representation Markup Language* was designed to describe the results of behavior planning, which are handed to realization [7]. MURML descriptions assume an incremental process model that synthesizes continuous speech and gesture in successive chunks. Each MURML specification contains (1) a textual definition of the verbal part of the utterance, possibly including internal chunk borders marked, and (2) specifications of paraverbal or nonverbal behaviors such as prosodic foci, gestures, or facial animations. MURML also focused on specifying the actual form of body or face behaviors. A communicative hand gesture is represented in terms of the morphological, spatio-temporal features of its meaningful phase (*wrist location* and *trajectory*, *hand shape*, *hand orientation)*, each of which being described either numerically or symbolically, building upon a notation system for sign languages. The overall structure of a gesture is given by defining *simultaneity*, *posteriority*, *repetitions*, or *symmetry* of those components. With regard to cross-behavioral temporal relations, the occurrence of a coverbal behavior can be defined either in terms of absolute times (start, end, duration) with regard to the start time of the chunk, or by simply stating the affiliation of the behavior with linguistic elements. Using time tags inserted in the text, behavior affiliations can be defined by referring to boundaries of co-expressive words. This way of specifying speech and coverbal behaviors separately allows partially overlapping temporal spans for behavior.

2.3 APML

The *Affective Presentation Markup Language* specifies the agent's behavior at the meaning level [4]. This language is based on Poggi's taxonomy of communicative functions which are defined as a pair (meaning, signal). Four different classes of communicative functions are differentiated depending on the type of information they convey: information about speaker's belief, goal, affective state and meta-cognitive information about speaker's mental state. A communicative function may be associated with different signals. That is for a given meaning, there may be several ways to communicate it. Another class of languages were designed to describe facial expression and gesture [4][5]. This separation of the languages ensures an independence between the mind module of the agent and the animation player.

2.4 RRL

The *Rich Representation Language* was developed in the NECA project (Net Environment for Emotional Embodied Conversational Agents). It focuses on presentation of simulated multimodal dialogue and was designed for representing the information that is relevant at the interfaces between components of a multimodal dialogue generation system[1] [11] that incrementally script a dialogue between two or more animated agents. The resulting RRL script remains independent of particular player technologies and can be mapped without adding new content to player-specific scripts.

An RRL document represents a dialogue at multiple levels of specification, ranging from an abstract dialogue plan level (scene generation) and an abstract verbal and non-verbal realizations of dialogue acts (multimodal natural language generation) to a concrete behavior specification temporally aligning phoneme-level information, facial expression and gesture. Each RRL document comprises four principal parts: (1) a representation of the initially shared information between the interlocutors (common ground); (2) the participants of the dialogue with name, gender, voice, or personality; (3) the dialogue acts along with their type, speaker, addressees, emotion category, semantic content, what it is a reaction to (adjacency pairs), and realization (prosody, gestures, and sentences out of words, syllables and phonemes); (4) the temporal ordering of the dialogue acts, specified as sequential or overlapping events. Underspecification is particularly useful for the relative timing of dialogue acts, for instance to specify that one multimodal utterance is followed by another, while a back-channel behavior of the listener starts with the first utterance.

[1] For the RRL XML Schema see www.ofai.at/research/nlu/NECA/RRL/index.html.

3 Towards a unified framework: SAIBA

The first step towards a unifying representational framework for multimodal generation is to lay down the general planning stages and knowledge structures that are involved in the creation of multimodal communicative behavior. We do not want to impose a particular micro-architecture here. Yet, as our goal is to define representation languages that can serve as clear interfaces at separate levels of abstraction—building upon our experiences from the abovementioned previous systems—we need to modularize the problem. We aim for the representation languages to be (1) independent of a particular application or domain, (2) independent of the employed graphics and sound player model, (3) and to represent a clear-cut separation between information types (function-related versus process-related specification of behavior).

The generation of natural multimodal output requires a time-critical production process with high flexibility. To scaffold this production process we introduced the *SAIBA* framework (Situation, Agent, Intention, Behavior, Animation), and specify the macro-scale multimodal generation consisting of processing stages on three different levels: (1) planning of a communicative intent, (2) planning of a multimodal realization of this intent, and (3) realization of the planned behaviors. See Fig. 1 for an illustration of the SAIBA framework.

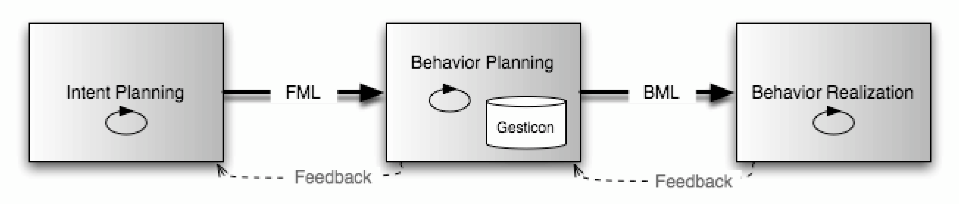

Fig. 1: SAIBA framework for multimodal generation.

The three levels lay down a general structure for every multimodal behavior generation system. While implemented systems have often concentrated on processing steps that pertain to one particular level, and have short-circuited others, we consider these subsequent stages to be in principle involved in the generation of each multimodal behavior an agent is to perform. The stages are bi-directionally linked to one another, with one stage delivering input to the next stage and feedback data running back to previous stages. Ideally, every stage along with its input and output representations is flexible and powerful enough to avoid limiting the expressiveness of the previous stage. We treat the processing within each stage and its internal structure largely as a "black box" or (more appropriately) as open research questions. Instead, we focus on the kind of data that is being processed at these stages, and on specifying the particular type and form of the information that needs to be represented as interfaces bridging the "gaps" between them. The rationale is that a clear-cut definition of information flow at the interfaces allows for a modular architecture and opens the possibility of combining solutions from different researchers without too much code modification.

The interface between stages (1) and (2)—Intent Planning and Behavior Planning—describes communicative and expressive intent without any reference to physical behavior. We call the language that we propose for specifying such information the *Function Markup Language* (FML). It is meant to provide a semantic description that accounts for the aspects that are relevant and influential in the planning of verbal and nonverbal behavior. An FML description must thus fulfill two functions. First, it must define the basic semantic units associated with a communicative event. Secondly, it should allow the annotation of these units with properties that further describe communicative function such as expressive, affective, discursive, epistemic, or pragmatic functions. Previous languages have started to address several of these issues, and a clearer picture of this terrain is forming, but coming up with a unified language is work in progress and beyond the scope of this paper.

The interface between (2) and (3)—Behavior Planning and Behavior Realization—describes multimodal behaviors as they are to be realized by the final stage of the generation pipeline. We propose the *Behavior Markup Language* (BML) for this purpose. In theory, a realization engine can realize every aspect of behavior (verbal, gestural, phonological, etc.) the behavior planner comes up with. In practice, when synthesizing and scheduling speech and animated graphics one often draws from a limited set of predefined animations or sound files. That is, the level of detail of representing behavior between behavior planning and behavior realization depends on the particular realization model. For instance, a realizer that employs a text-to-speech module and is able to produce movements on the fly by means of, e.g., procedural animations could take as input rather flexible descriptions of single words along with prosodic commands or the morphological features of a hand gesture. A behavior realization that rests upon a fixed repository of animations and allows for a low degree of parameterization would need a unique identifier along with a set of appropriate parameters. We aim for BML to stay above such specific process implementations (the boxes in Figure 1), i.e. to provide a general, player-independent description of multimodal behavior that can be used to control an agent. Nevertheless, it needs to provide a sufficient level of detail in describing behavior, from the mere occurrence and the relative timing of the involved actions, to the detailed (yet player-independent) definition of a behavior's form. Behavior Planning will thus be concerned with fleshing out a BML description in necessary detail. In concretizing this specification, the planner could draw upon a lexicon of BML behavior definitions, a *Gesticon* (see Figure 1), that would also provide a basis for attuning to the capabilities of the realizer. Further, it is possible that multiple lexicons like the Gesticon are used by the processes at each stage of planning. This choice is dependent on the particular approach and architectural use of the SAIBA model.

4 Behavior Markup Language: BML

This section describes the proposed communicative behavior markup language, starting with the general features of the language that address fundamental requirements and then goes on to describe some of the behaviors that will be covered. It should be pointed out again that this is work in progress and therefore BML will continue to evolve as our collaboration matures.

4.1 General Structure

The communicative behavior markup language, or BML, is an XML based language that can be embedded in a larger XML message or document simply by starting a `<bml>` block and filling it with behaviors that need to be realized by an agent. The behaviors are listed one after another, at the same level in the XML hierarchy, with no significance given to their order. Generally the behaviors are single elements that contain no text or other elements, but this is not required. Behavior parameters, some of which are general and some of which are behavior specific, are specified as attribute values of the behavior element. A simple behavior block is shown in Fig. 2.

```
<bml>
  <head id=”h1” type=”nod” amount=”0.4”/>
  <face id=”f1” type=”eyebrows” amount=”1.0”/>
</bml>
```

Fig. 2: A simple example of a BML block.

Most attributes in a behavior element are optional and the assumption is that reasonable default values will be used when attributes are left out. For example, the behavior `<head type=”nod”/>` could be expected to produce a typical nod. If no timing constraints are given, the behaviors are all expected to start immediately and run for their default durations.

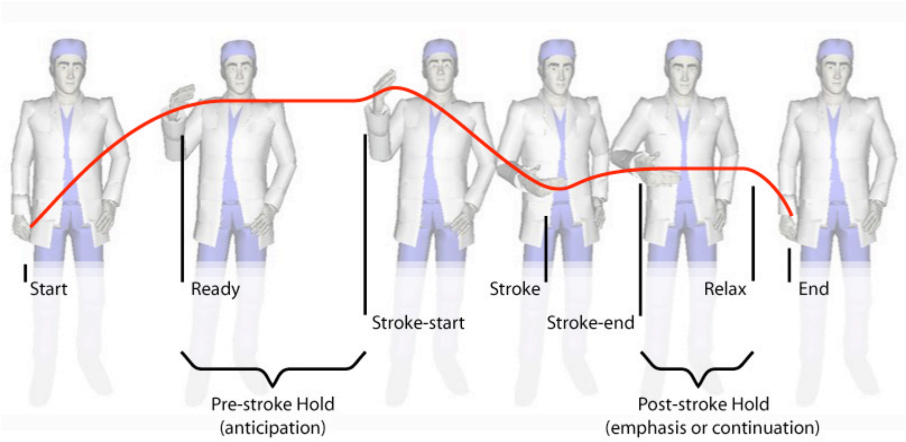

Fig. 3: The synchronization points of a communicative behavior.

4.2 Synchronization

While some meaning can be carried by the form of a communicative behavior, it’s also the temporal context that shapes its interpretation. For example, the co-occurrence of a pointing gesture and the verbalization of the phrase “that one”, allows us to locate a unique referent. Similarly, seeing someone express disgust on their face as they gaze upon their food, gives us a strong clue about what the emotional display describes. In addition to co-occurrence, the order and in-between timing can also demonstrate meaningful relationships. Therefore an important feature of BML is the specification of temporal constraints. When the communication of a particular intent relies on timing, the behavior planner needs to fill in crucial timing constraints while leaving all other timing information unspecified. This gives the realizer maximum flexibility for realizing the behavior while ensuring that meaning does not get lost.

Synchronization Points. Temporal constraints in BML are specified using two important constructs: A behavior ID and a behavior synchronization point. The behavior ID is a unique identifier given to a behavior instance so that it can be referenced from other behaviors. The behavior synchronization points, or sync-points, provide significant points of alignment between behaviors based on the typical movement phases that a behavior goes through during its realization. These phases are illustrated in Fig. 3.

The preparation for or visible anticipation of the behavior occurs between *start* and *ready*, and the retraction back to neutral or previous state occurs between *relax* and *end*. The actual behavior takes place between the *ready* and *relax*, with the most significant or semantically rich motion during the stroke phase, between *stroke-start* and *stroke-end,* with the greatest effort coinciding with the *stroke* point. A behavior does not need to have a stroke phase, so for example looking at something will only involve ready (the time of making eye-contact) and relax (the time of breaking eye-contact). If no preparation or relaxation is needed, then start and ready refer to the same point in time, and relax and end refer to the same point in time.

```
<bml>
  <gesture id=”g1” type=”beat”/>
  <head type=”nod” stroke=”g1:stroke”/>
  <gaze target=”object1” start=”g1:ready” end=”g1:relax”/>
</bml>
```

Fig. 4: An example of synchronizing head movement and gaze with a gesture.

The sync-points are actual attributes in all BML behavior elements and their value can be a reference to any other sync-point, ensuring temporal alignment. A simple example of a head nod and a targeted gaze co-occurring with various phases of a gesture is given in Fig. 4.

Conditions and Events. In addition to aligning sync-points of behaviors with each other, they can also be aligned with a sync-point that gets triggered based on some condition or by the arrival of an event. This is accomplished by introducing a special `<wait>` behavior whose duration is either determined by the satisfaction of a condition or the reception of an event. As a fall-back, a time-out duration can also be specified. A couple of different examples of how `<wait>` can be used are given in Fig. 5. The second example also introduces an `<event>` behavior that can generate events for synchronization.

4.3 Behavior Elements

The behaviors themselves fall into general behavior categories that can then be further defined through a possible sub-type attribute and several type-specific attributes. The general behavior categorization is meant to be fairly stable, while the set of attributes are expected to evolve with ongoing research (see Table 1). Not all attributes are required, and some attributes may refine the behavior in such detail that only certain animation system can make use of them. The particular decomposition is motivated on the one hand by high-level considerations such as a) physiology (muscular contraction and joint articulation), and b) existing studies on communicative non-verbal behavior. On the other hand there are computational factors. For instance: the same hand configuration can be used with several arm movements. Gaze and head movement are separated in order to provide more flexibility for animation of gaze behavior, in particular allowing head movement while gazing at something. In our formal specification we use `<torso>` for characterizing spine movement and shoulders. While in the `<body>` element pelvis and legs are specified as parts of posture. The naming of behavior elements is mostly drawn from the existing set of XML languages discussed earlier in this paper.

Each of these BML elements contains attributes that describe the visual appearance and movement dynamics of the behavior in order to achieve certain expressive effects. In what follows we briefly describe gesture, gaze and face behaviors since they are commonly used during the communicative process and often get special attention in ECAs systems. All of them show different kinds of complexity.

BML Element	Description
`<head>`	Movement of the head independent of eyes. Types include nodding, shaking, tossing and orienting to a given angle.
`<torso>`	Movement of the orientation and shape of the spine and shoulder.
`<face>`	Movement of facial muscles to form certain expressions. Types include eyebrow, eyelid and larger expressive mouth movements.
`<gaze>`	Coordinated movement of the eyes, neck and head direction, indicating where the character is looking.
`<body>`	Full body movement, generally independent of the other behaviors. Types include overall orientation, position and posture.
`<legs>`	Movements of the body elements downward from the hip: pelvis, hip, legs including knee, toes and ankle.
`<gesture>`	Coordinated movement with arms and hands, including pointing, reaching, emphasizing (beating), depicting and signaling.
`<speech>`	Verbal and paraverbal behavior, including the words to be spoken (for example by a speech synthesizer), prosody information and special paralinguistic behaviors (for example filled pauses).
`<lips>`	This element is used for controlling lip shapes including the visualization of phonemes.

Table 1: The BML behavior elements.

Gesture entry. Gestures are complex, usally being composed by one or a sequence of basic gesture elements, each of which describes a basic hand-arm movement trajectory. Adapted from MURML and ASL hand shape configuration description [13], gestures are composed of trajectory, hand shape, a thumb orientation, and fingers shapes. Complex gestures are represented by means of a collection of behavior elements with different type attributes, and which are aligned via synchronization points.

Face entry. Within behavior specification languages facial expressions are often described by a set of labels such as smile, raise eyebrow, open mouth etc. Such descriptions limit the encoding variability. While FACS allows variability in the specification of surface form, it is not widely used in the graphics community. Facial expression in graphics models is commonly described by sets of low-level parameters, e.g. MPEG-4, or via muscle contraction. Thus to be independent of individual facial models, we propose describing facial expression via sets of face elements, with each set being a placeholder for various model-dependent facial descriptions. Via synchronization points we are able to account for the two major approaches to facial display of emotions: (1) The more static and traditional one, where whole emotional displays are switched on and off instantaneously and (2) the more dynamic approach where emotional displays gradually set in and slowly fade out.

```
<bml>
  <gesture id=”g1” type=”point” target=”object1”/>
  <body id=”b1” posture=”sit”/>
  <wait id=”w1” condition=”g1:end AND b1:end”/>
  <gaze target=”object2” start=”w1:end”/>
</bml>

<bml>
  <speech id="s1" type="text/plain">
      First sentence.
  </speech>
  <event start=”s1:end” emit="ACT1_COMPLETE" />
</bml>

<bml>
  <wait id="w1" event="ACT1_COMPLETE" duration=”5.0”/>
  <speech type="text/plain" start="w1:end">
      Second sentence.
  </speech>
</bml>
```

Fig. 5: Examples of how `<wait>` can align a behavior with a condition or an event.

Gaze Entry. Gaze is another example of a complex modality, comprising: (1) only eye direction, (2) neck, head and eyes showing one direction or (3) neck, head and eyes showing individual directions. Via the referencing mechanism of BML, gaze direction is specified relative to a target. This is different from FACS and MPEG-4 where gaze direction is absolute (e.g. defined by angle values).

Speech Entry. The speech element is used for specifying the verbal and paraverbal behavior. It typically contains marked-up text to be rendered by a speech synthesizer but also may contain references to plain sound files. The purpose of the mark-up is two-fold. On the one hand it is used for supporting the synthesis process by giving directives on prosody and pronunciation; on the other hand the mark-up is used for identifying elements (e.g. words or prosodic boundaries) within the text. These elements are then to be used as references for the synchronization of speech and non-verbal behavior. In order to keep BML flexible enough to deal with the considerable variety of existing speech synthesizers and speech markup languages (e.g. SSML or VoiceXML) the actual type of mark-up language is left open.

4.4 Gesticon

The design of BML allows for disentangling a behavior *form description* and its *instantiation* in a communicative process. It is therefore easy to create what has been called a Gesticon [9] or Behavior Lexicon [17] – a dictionary of behavior descriptions. A strong property of the BML language is its independence of graphics models, rendering technologies and applications. Creating gesture shape, facial expression, body posture etc. can be very time consuming, so sharing behavior definitions would be a great help to the ECA community.

5 Usage

As we have already mentioned, the Behavior Markup Language presented here is an improvement and extension of prior related languages that have been used extensively in many projects [2, 3, 4, 7, 9, 11, 12, 17, 20]. The authors and their labs have committed to moving towards BML in their work, and several efforts are already underway to build parsers and planning modules that are compatible with it. In the spirit of this effort we are hopeful that software will be made available under relatively open licenses that enables others to adopt the BML in their work without having to replicate a lot of the work that has gone into the language itself as well as the software modules that can use it. Since the processes are intentionally left unspecified as open research questions, we envision there being a selection of approaches to how BML (and later FML) are used and produced – this is where the benefits of a unified approach will become even clearer: By leaving out a reference to the processes that produce and control behavior we intend to reap the benefits of a common foundation without closing the door on new methods of planning and control.

Besides working toward the actual employment of BML and FML in our own system to accelerate the development of software modules, we think it is central to inform and guide the development of the overall framework by working towards differently challenging scenarios. We have thus started to define use cases that describe the kinds of natural multimodal behavior that we ultimately want to be able to specify in BML and FML. Such cases include (1) a speaker waiting until she has finished pointing at an object and has determined that the listener is also attending to the object; (2) two agents shaking hands; (3) a speaker saying "Give me [the cake]", where [the cake] is accompanied by an iconic gesture describing an attribute of the cake; (4) a speaker using indirectness as part of politenss, such as saying "how are we doing for time" to indicate "we need to hurry". Use cases like these pose specific and big challenges for representing and generating multimodal behavior. While implementing and extending our framework, also beyond what we aim to achieve in our current research settings, we consider using these test cases as milestones for actual demos.

6 Conclusion

In this paper we have presented first steps toward a common framework for multimodal generation, enabling people to better work together and to use each other's work with a minimal amount of custom effort. We have proposed BML as a representation language meant as a clear interface between modules of behavior planning and realization. The current focus of this work has been to specify the communicative and expressive behaviors traditionally associated with explicit, verbal communication in face-to-face dialog. We plan to explore extensions to BML and FML that support additional kinds of behavior, including those that may not have any associated communicative intent. Specifying such behaviors may lead us to a more general scripting language that incorporates the BML work described here as a key component. We encourage all researchers and practitioners working on intelligent virtual agents to contribute to this ongoing effort. The more people that join this discussion, the better the chances that we will find a representation that provides the groundwork for many of the employed generation systems, and where people can actively collaborate either by sharing their experiences and knowledge or by directly exchanging system components. The *Mindmakers* website (www.mindmakers.org) provides a forum for discussion and collaboration related to this effort, as well as documentation of the full BML specification.

Acknowledgements

We would like to thank our other collaborators in this effort, in particular Lewis Johnson and Norm Badler. We are grateful to Reykjavik University for supporting our meeting in Iceland. The work on the Gesticon has been partially supported by the EU Network of Excellence HUMAINE (IST-507422) and by the Austrian Funds for Research and Technology Promotion for Industry (FFF 808818/2970 KA/SA). CADIA participation was made possible in part by RANNÍS grant 050013021 and a Marie Curie European Reintegration Grant within the 6th European Community Framework Programme. OFAI is supported by the Austrian Federal Ministry for Education, Science and Culture and by the Austrian Federal Ministry for Transport, Innovation and Technology. This publication reflects only the authors' views. The European Union is not liable for any use that may be made of the information contained herein.

References

1. Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Douville, B., Prevost, S., and Stone, M., (1994) Animated Conversation: Rule-Based Generation of Facial Expression, Gesture and Spoken Intonation for Multiple Conversational Agents. Siggraph 94 Conference Proceedings, ACM SIGGRAPH, Addison Wesley, 413-420.
2. Cassell, J., Vilhjálmsson, H., and Bickmore T., (2001) BEAT : the Behavior Expression Animation Toolkit, Proc. ACM SIGGRAPH 2001, Los Angeles, August 12-17, 477-486.
3. Cassell, J., Vilhjálmsson, H., Chang, K., Bickmore, T., Campbell, L. and Yan, H., (1999). Requirements for an Architecture for Embodied Conversational Characters, Computer

Animation and Simulation '99 (Eurographics Series). Vienna, Austria: Springer Verlag

4. DeCarolis B., Pelachaud C., Poggi I, and Steedman M. (2004). APML, a mark-up language for believable behavior generation. In H. Prendinger and M. Ishizuka, editors, *Life-like Characters. Tools, Affective Functions and Applications*, 65--85. Springer.
5. Hartmann, B., Mancini, M., and Pelachaud, C. (2002). Formational parameters and adaptive prototype instantiation for MPEG-4 compliant gesture synthesis. In *Computer Animation'02*, Geneva, Switzerland. IEEE Computer Society Press.
6. Kopp S.,, B. Jung, N. Lessmann, I. Wachsmuth:, (2003) Max--A Multimodal Assistant in Virtual Reality Construction. KI 4/03: 11-17.
7. Kopp S., Wachsmuth I. (2004): Synthesizing Multimodal Utterances for Conversational Agents. *Computer Animation and Virtual Worlds,* 15(1): 39-52
8. Krenn B. (2005). Representational Lego for ECAs, *Background paper for a presentation held at the FP6 NoE HUMAINE Workshop on Emotion and Interaction. Paris, 10-11 March*.
9. Krenn B., Pirker H. (2004). Defining the Gesticon: Language and Gesture Coordination for Interacting Embodied Agents, in *Proc. of the AISB-2004 Symposium on Language, Speech and Gesture for Expressive Characters*, University of Leeds, UK, 107-115.
10. Martell C. (2002). FORM: An Extensible, Kinematically-based Gesture Annotation Scheme in *Proceedings of the 2002 International Conference on Language Resources and Evaluation*, Las Palmas, Canary Island.
11. Piwek P., Krenn B., Schröder M., Grice M., Baumann S., Pirker H. (2002). RRL: A Rich Representation Language for the Description of Agent Behaviour in NECA, In Proceedings of the Workshop *Embodied conversational agents - let's specify and evaluate them!,* held in conjunction with AAMAS-02, July 16 2002, Bologna, Italy.
12. Prendinger H., Descamps S., Ishizuka M. (2004). MPML: A markup language for controlling the behavior of life-like characters, *Journal of Visual Languages and Computing*,15(2):183-203
13. Stokoe W. C., Casterline D. C. and Croneberg C. G. (1976). *A dictionary of American sign language on linguistic principles*. Linstok Press
14. Searle, J. R. (1969). *Speech acts: An essay in the philosophy of language*. London: Cambridge Univ. Press.
15. Thórisson, K. R. (1993). Dialogue Control in Social Interface Agents. *InterCHI Adjunct Proceedings '93*, Amsterdam, April, 139-140.
16. Thórisson, K. R. (1995). Computational Characteristics of Multimodal Dialogue. *AAAI Fall Symposium on Embodied Language and Action*, Massachusetts Institute of Technology, Cambridge, Massachusetts, November 10-12, 102-108.
17. Thórisson, K. R. (1999). A Mind Model for Multimodal Communicative Creatures and Humanoids. International Journal of Applied Artificial Intelligence, 13(4-5): 449-486.
18. Thórisson, K. R., Vilhjalmsson, H., Kopp, S., Pelachaud, C. (2006). Report on Representations for Multimodal Generation Workshop. *AI Magazine*, 27(1), 108.
19. Vilhjalmsson, H. (2004). Animating Conversation in Online Games. In M. Rauterberg (ed.), *Entertainment Computing ICEC 2004, Lecture Notes in Computer Science 3166*, 139-150, Springer.
20. Vilhjalmsson, H. (2005). Augmenting Online Conversation through Automated Discourse Tagging, 6th annual minitrack on Persistent Conversation at the 38th Hawaii International Conference on System Sciences, January 3-6, Hilton Waikoloa Village, Big Island, Hawaii, IEEE.