The Behavior Markup Language: Recent Developments and Challenges

Hannes Vilhjálmsson[1], Nathan Cantelmo[2], Justine Cassell[2], Nicolas E. Chafai[3], Michael Kipp[4], Stefan Kopp[5], Maurizio Mancini[3], Stacy Marsella[7], Andrew N. Marshall[7], Catherine Pelachaud[3], Zsofi Ruttkay[6], Kristinn R. Thórisson[1], Herwin van Welbergen[6], and Rick J. van der Werf[6]

[1] CADIA, Reykjavik University, Iceland
{hannes,thorisson}@ru.is
[2] ArticuLab, Northwestern University, USA
{n-cantelmo,justine}@northwestern.edu
[3] IUT de Montreuil, University de Paris 8, France
{n.chafai,m.mancini,pelachaud}@iut.univ-paris8.fr
[4] DFKI, Germany
Michael.kipp@dfki.de
[5] Artificial Intelligence Group, University of Bielefeld, Germany
skopp@techfak.uni-bielefeld.de
[6] Human Media Interaction, University of Twente, The Netherlands
{z.m.ruttkay,h.vanwelbergen,werf}@ewi.utwente.nl
[7] Information Sciences Institute, University of Southern California, USA
{marsella,amarshal}@isi.edu

Abstract. Since the beginning of the SAIBA effort to unify key interfaces in the multi-modal behavior generation process, the Behavior Markup Language (BML) has both gained ground as an important component in many projects worldwide, and continues to undergo further refinement. This paper reports on the progress made in the last year in further developing BML. It discusses some of the key challenges identified that the effort is facing, and reviews a number of projects that already are making use of BML or support its use.

1 Introduction

The goal of the SAIBA effort is to create a representational framework for real-time multimodal behavior generation in embodied conversational agents. Its members have proposed knowledge structures that describe the form and generation of multimodal communicative behavior at different levels of abstraction. The levels represent the interfaces between the stages for (1) planning communicative intent, (2) planning multimodal realization of this intent, and (3) the realization of the planned behaviors (see Fig. 1). Mediating between the first two stages is the Functional Markup Language (FML) that describes intent without reference to surface form; the FML still remains largely undefined. Between the last two stages sits the Behavior Markup Language (BML) that describes human nonverbal and verbal behavior in a manner independent of the particular realization (animation) method used.

C. Pelachaud et al. (Eds.): IVA 2007, LNAI 4722, pp. 99–111, 2007.

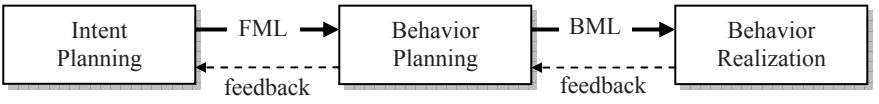

Fig. 1. The three stages of behavior generation in the SAIBA framework and the two mediating languages FML and BML

This is an ongoing effort that was officially kicked off when a group of international researchers in the area of multimodal communication and computer animation came together at Reykjavik University in April 2005. A paper published at IVA in 2006 introduces the framework and describes the first iteration of the Behavior Markup Language [5] and a wiki site hosted by the Mindmakers community portal tracks its development[1]. An active discussion forum and a mailing list are linked off the wiki site and we invite all interested to join and participate in the discussions. The material in this paper is mainly drawn from the most recent workshop which was hosted by OFAI in Vienna, Austria, in November 2006. The next workshop on BML will take place in Paris, France, in June 2007.

The paper is organized as follows: First we give a quick overview of BML in section 2 and then focus on some of the recent developments in section 3. Noteworthy challenges that we have identified are described in section 4. A review of a range of research projects that are pioneering the use of BML is provided in section 5 and finally some conclusions wrap up the paper in section 6.

2 Quick Overview of BML

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 should be realized by an animated agent.

For example:

```
<bml>
   <speech id="s1" type="application/ssml+xml">
     <text>This is an <mark name="wb3"> example</text>
   </speech>
   <head id="h1" type="NOD" stroke="s1:start"/>
   <gesture id="g1" stroke="s1:wb3" relax="s1:end" type="BEAT">
     <description level="1" type="MURML">...
     </description>
   </gesture>
   <gaze id="z1" target="PERSON1" stroke="g1:stroke-0.1"/>
   <body id="p1" posture="RELAXED" start="after(s1:end)"/>
   <cadia:operate target="SWITCH1" stroke="p1:ready"/>
</bml>
```

[1] http://wiki.mindmakers.org/projects:BML:main

This block coordinates speech, gesture, gaze, head and body movement by including a set of corresponding behavior elements inside a single `<bml>` element. Other possible behavior elements include torso, face, legs, lips and a wait behavior. Every behavior is divided into six animation phases. Each phase is bounded by a *sync-point* that carries the name of the motion transition it represents. The seven sync-points are: *start*, *ready*, *stroke-start*, *stroke*, *stroke-end*, *relax* and *end*. Synchrony between behaviors is achieved by assigning the sync-point of one behavior to a sync-point of another, causing the two to align at that point in time. In the example above, the stroke (most effortful part) of the head nod occurs exactly when the speech starts. An SSML annotation of the text to be spoken generates a new sync-point called *wb3*, which the gesture aligns its stroke with. The gaze uses a negative offset of 100 ms to make sure it rests on its target named PERSON1 just before the stroke of the gesture.

The body behavior uses a new feature to indicate that it can occur any time after the speech ends. This feature and other recent synchronization developments are discussed further in section 3.3. An important recent addition to BML is levels of description, which provide a way to describe a behavior in more detail than is possible with the core set of BML tags and attributes (see Fig. 2). Here a MURML type description of the gesture is included (not shown in full) with the core BML gesture element. Section 3.1 will explain the new levels of description in more detail. In order to introduce candidates for core BML extensions, namespaces can be used, as demonstrated by the hypothetical *operate* behavior above which is qualified with a CADIA specific namespace. This feature is covered in section 3.2.

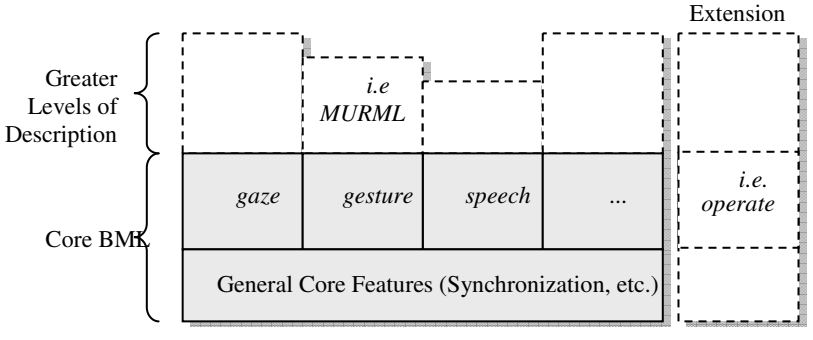

Fig. 2. The core BML specification of a behavior can be further refined through greater levels of description, while namespaces can provide general extensions

3 Recent Developments

3.1 Levels of Description

When describing behavior in great detail for animation, it makes good practical sense to use whatever advanced capabilities the character animation engine might support, such as a particularly high level of articulation or a simulation of physical dynamics. Furthermore, a behavior planner may want to generate behavior at multiple levels of detail so that a behavior realizer can adjust the required realization effort based on

some chosen level of detail in the environment. Just as it would make little sense to cover all interaction over the Internet in a mail protocol, it would be an impossible task to cram the entire range of parameters required to address a variety of advanced animation methods and levels of detail into the core BML specification. One reason is that without a very specific goal, it is difficult to select between the many options for describing a particular behavior at a particular level of detail. One way is to put a large number of alternatives into the same language, but then we would quickly end up with a bloated specification language, trying to be everything to everyone and ending up being abandoned by all.

Instead of this route, it was decided to introduce levels of description to BML and define a core BML specification that would be kept completely independent of animation engines and developed to strike a good balance between being expressive and being lean. Additional levels can include a more detailed or a more engine specific description of the behavior already described in core BML. It is still an objective to allow re-use at the other description levels and the aim is to see the emergence of a set of description methods corresponding to some of the major animation techniques. Whether we can achieve a unified ontology of detail across multiple levels remains to be seen, but to begin with we can see the various descriptions as kinds of "plug-ins" for the various techniques.

The core BML description corresponds to the basic behavior elements and attributes introduced in [5] and the online reference[1]. Additional levels are embedded within a behavior element as an XML *description* element that can contain arbitrary XML. The type attribute of the description element should identify the type of content, indicating how it should be interpreted and executed by the behavior realizer.

This example shows BML describing gaze towards another character called PERSON1. *The attributes inside the* <gaze> *element are part of core BML while the embedded description levels of type* RU.ACT *and* ISI.SBM *(names identifying projects) provide additional parameters.*

```
<bml id="bml1">
  <gaze id="gaze1" target="PERSON1">
    <description level="1" type="RU.ACT">
      <target>PERSON1</target>
      <intensity>0.6</intensity>
      <lean>0.4</lean>
    </description>
    <description level="2" type="ISI.SBM">
      ...
    </description>
  </gaze>
</bml>
```

If additional levels of description cannot be interpreted by a realizer, it needs to be able to fall back on the core BML which should still provide a reasonable, if a somewhat simpler, description. Therefore, core BML needs to be present, even if higher levels of description make it redundant. This is crucial for the portability of a behavior planner, since you can't know for sure what levels beyond the core other realizers support.

All BML-compliant behavior realizers have to guarantee that they can interpret core BML behavior descriptions and display them correctly. In those cases where a realizer is only providing a special subset of BML, for example a talking head, it should be made very clear and behaviors that are not realized should produce appropriate feedback messages (see 3.4). Those realizers that can interpret any of the higher levels of description should make use of those. If a realizer is expecting a description of a certain level but does not receive it, it should default to the core description.

Numbered description levels imply a particular order, and generally higher numbers can be associated with richer descriptions. In that sense, the numbers can be seen either as a priority, where richer descriptions would be picked before poorer ones, or as roughly corresponding to a level of detail. However, even if behavior planners request a performance at a certain description level, a realizer may end up picking a description based on its own capabilities (sending a warning message back to the behavior planner). The overall effort will now focus on solidifying the first version of the core BML specification and allow working groups to discuss other levels of description pertinent to their specific research interests.

3.2 Namespace Extensions

The core BML behavior elements will continue to grow with new versions of the specification because the ongoing work behind BML involves identifying and defining a broad and flexible library of behaviors. Implementers are encouraged to explore new behavior elements and specialized attributes when making use of BML. Any such experimental components that cannot be embedded within a special level of description, should be identified as non-standard BML by utilizing XML namespaces to prefix the elements and attributes.

This example utilizes customized behaviors from the Smartbody project. Here, we use the namespace sbm *(short for SmartBody Module).*

```
<bml>
  <sbm:animation name="CrossedArms_RArm_beat"/>
  <gaze target="PERSON1"
      sbm:joint-speeds="100 100 100 300 600"/>
</bml>
```

3.3 Special Synchronization Features

BML provides special sync-points with every behavior block, *bml:start* and *bml:end*, that refer to the start of the earliest behavior in the request and the end of the latest behavior. Aligning to *bml:start* and *bml:end* requires special precautions. If there is no offset specified, only *start* sync-points can be aligned to *bml:start*, and only *end* sync-points can be aligned to *bml:end*. If there is an offset specified, it must be positive when referring to *bml:start* and negative when referring to *bml:end*. This constraint ensures that *bml:start* and *bml:end* properly point at the very start and end of the behavior request.

Aligning sync-points provides exact timing information, but there are times when a behavior planner only cares about the order in which behaviors occur and not so much about the exact moments in time they occur. To allow for this flexibility BML now allows under specified timing constraints in the form of the predicates *before()* and *after()*. The time *before(sync-point)* occurs at any point before the indicated sync-point and the time *after(sync-point)* occurs at any point after the indicated sync-point.

3.4 Feedback

It is important that the behavior planner get information back from the behavior realizer about the success or failure of various behavior requests in order to successfully plan subsequent action. This is particularly true for real-time systems that incrementally generate behavior.

Three main kinds of feedback have been identified and have received a special element in the core specification, while the particular system-level messaging protocol is not being defined by BML. The first kind is the regular `<event>` message, which can be scheduled to be emitted upon successfully reaching any sync-point in a behavior block and is automatically sent when an entire block successfully finishes. A special `<warning>` message is sent whenever the behavior realizer is unable to 100% comply with a behavior request, but was able to work within permissible soft boundaries. For example, a warning is issued if a soft timing constraint could not be honored or a behavior could not be performed at a recommended level of description. Finally, an `<exception>` message is returned when a behavior or a block of behaviors had to be cancelled because hard constraints could not be honored or because of an interrupt or other kinds of realization problems.

4 Open Challenges

4.1 Maintenance of Behavior

If one thinks about behavior as something that has a fixed duration and does not have a lasting impact of any kind, it is straight-forward to keep track of what the body is doing. At any given time, one can simply sum up all the behaviors currently executing to see exactly what is going on. However, reality is not that simple. When a behavior command is completed, the body is typically left in a new state, possibly even maintaining the behavior until another behavior replaces it.

One example is the gaze behavior. If a character is asked to gaze at a certain target with the command `<gaze target='person1' stroke='g1:stroke'/>`, it is clear that the gaze will fully rest on the target at exactly the same time another behavior (g1) reaches its own moment of greatest effort. However, it is completely left undetermined what happens next. Does the character continue to look at *person1*? If *person1* starts moving, should the character adjust its gaze accordingly? If the character is being requested to track *person1* with its gaze, how should that be indicated and how long should it last? What would then happen if *person1* left the scene? What if during tracking there was request for a short glance at the watch – would the gaze return to tracking *person1* after the glance? If not, where should the gaze go after looking at the watch?

Clearly there is a lot left undetermined here and the behavior realizer has a great deal of flexibility to fill in the missing pieces. However, it is reasonable for a behavior planner to want to know what happens between behavior requests or even to be able to describe what should happen. One way to approach this is for the behavior planner to describe to the realizer (a) rules for maintaining behaviors, such as "keep doing this until I tell you otherwise" and (b) default behavior states that the body should return to when other behaviors run their course. No structure to do this is part of BML yet, except for a few special behaviors that assume continuous maintenance like `<body pose='sit'/>`. Another way is to have a much tighter command-feedback cycle where the planner takes charge of all decisions and continuously reacts to events from the world and from the realizer. That approach is already possible with BML and needs to be maintained as an equally valid approach.

4.2 Constraints and Synchronization

The synchrony achieved through aligning behavior sync-points demonstrates one type of timing constraint, but it is important to consider other types as well. These include:

- *Physical characteristics constraints:* Limitations on body movement speed.
- *Rhythmic constraints:* Requirement that different modalities stay in perfect synchrony.
- *Global rule constraint:* A general synchrony rule that is meant to hold true for all behaviors in a set of behaviors (such as stating that all gesture strokes should coincide or precede emphasized syllables).
- *Fixed signal constraint:* Synchronization to an external source with a fixed timing pattern (such as clapping to music).
- *Perceptual signal constraint:* Synchronization to an external signal with unknown timing (such as the actions of another person).

While it is evident that all these kinds of timing constraints 'make sense' for Virtual Humans, they raise some challenges for the SAIBA framework:

- *Where, and how should the different constraints be declared?* Should their description be deferred to FML or should specification of character state for example become part of BML?
- *What type of constraints are necessary, as of the numerically and of strength?* As of the nature of the constraints, the linear interval constraints (on sync-points of gestures), similar to ones used for facial expressions seem to be expressive enough. Two categories are needed though: hard and soft constraints. The latter might need ranks, indicating how important a certain constraint is. Another option would be to consider the precision of constraints: even if a gesture cannot be scheduled to the required time, effort could be made to schedule it 'as close as possible'.
- *Where and how can they all be examined together for making final timing decisions ?* A behavior planner or a behavior realizer may need to collect all timing constraints from various sources for combining redundant ones, resolving possible conflicts and for finally providing a flattened uniform representation for correct and efficient temporal scheduling.
- *What solution principles should be used to cope with under-constrained and over-constrained cases?*

The declaration and manipulation of constraints is related to levels of description. If only core BML gestures are available from a particular gesture repository, it is possible to omit the prescription of constraints that assure subtle variants of those gestures. A conceptually clear and computationally efficient way to handle all these possible constraints is needed and is currently being explored.

5 Current Projects and Tools

This section reviews a few projects that incorporate BML. These projects are tentatively classified into: Full ECA systems, behavior planners, behavior realizers, repositories and tools. This is informal, and only meant to demonstrate the range of projects and how they potentially fit together in the SAIBA framework (see Fig. 1).

5.1 Full ECA Systems

RVT: The Reactive Virtual Trainer[2]. The *Reactive Virtual Trainer* (RVT) [10] is an ECA capable of presenting physical exercises that are to be performed by a human, while monitoring the user and providing feedback. The reactivity of the RVT is manifested in natural language comments, readjusting the tempo, pointing out mistakes or rescheduling the exercises. These exercises can be performed to the beat of a user's favorite music. Exercises describe a mix of behaviors in different modalities, including exercise movement, sound (such as clapping, feet tapping), speech and music. The main extension to core BML is the *observer* behavior. The observer is used to provide coordination with outside world events that are predictable and possibly repeat themselves, such as the beat in music. Such synchronization can not be achieved by the event/wait system described in core BML since BML events are, by design, non-repeatable and unpredictable.

This example shows counting at the beat of beatObserver1.

```
<bml>
  <bmlt:observer id="beatObserver1"/>
  <speech id="s1" start="beatObserver1:1">One</speech>
  <speech id="s2" start="beatObserver1:2">Two</speech>
  <speech id="s3" start="beatObserver1:3">Three</speech>
</bml>
```

Ambulation Agents[3]. *EVE-Online* is a massively multiplayer online role playing game (MMORPG) where over 30.000 simultaneous players travel through a vast galaxy in custom configured ships, seeking fame and fortune through exploration, trade, conflict and political power. So far, players have been bound to their ships, but the developer of this game, CCP Games, has decided to develop an addition to this game called *Ambulation* that allows players to exit their ships at space stations and mingle with other players and fully autonomous agents, represented by animated graphical avatars. To achieve believable interactions between characters, this project will rely on automating the coordination of nonverbal social behavior. A special

[2] Developed at Human Media Interaction, University of Twente.
[3] Developed jointly by CADIA, Reykjavík University and CCP Games.

version of the game client is being equipped with a BML interface, so that behavior planning can be fully implemented within the SAIBA framework, both ensuring CCP access to existing domain expertise and ensuring that new results from this project will immediately be available to the research community. It is expected that this work will particularly contribute to BML by exploring support for tight perception-action loops in a highly dynamic social environment.

SuperRadioHost[4]. *SuperRadioHost* is an autonomous radio show host designed to create a radio program from scratch and execute it in real-time – on the air – including creating song introductions and conducting interviews with real people. The system perceives its own on-air transmissions including its own speech content and prosody. For interviews it employs a new model of turntaking that goes well beyond what has been built in the past [11]. The system's planning mechanisms embody a version of the SAIBA framework and propose a mixture of opportunism and predefined plans. Naturally the system only uses speech-related markup for describing its behaviors. Among the key research topics addressed is opportunistic planning in real-time dialogue and holistic models of cognitive agents.

5.2 Behavior Planners

NVB: Non-verbal Behavior Generator[7]. The *Non-Verbal Behavior Generator* (NVB) [7] is a rule-based behavior planner, inspired by BEAT [1], that analyses a virtual human's communicative intent, emotional state, and text (dialog) and generates appropriate nonverbal behaviors. The inputs are described using a preliminary version of the FML format and the final behavior commands are specified in BML, which are passed to *SmartBody*. The NVB generator is designed for portability and for being imported into different virtual human systems with different animation schemes by simply changing the names of the behaviors in the behavior description repository without modifying the nonverbal behavior rules. NVB is used in the SASO-ST, ELECT and Virtual Patient systems, among others.

NOVA: Nonverbal Action Generator[5]. NOVA is a system able to recreate the gesture behaviour of a specific human performer using statistical models, a fixed repertoire of gestures and procedural animation [8]. It is intended as an alternative/complement to rule-based approaches to generate a varied and natural looking "baseline" behaviour. With respect to SAIBA, we plan to (1) translate the descriptions of our gesture repertoire to emerging BML descriptors and (2) make our planner communicate to the realizer using BML. A central research challenge for NOVA will be the merging of two BML streams to combine our probabilistic approach with rule-based ones.

5.3 Behavior Realizers

ACE: The Articulated Communicator Engine[6]. The *Articulated Communicator Engine* (ACE) is a behavior realization engine that allows the modeling of virtual

[4] Developed at CADIA, Reykjavík University.
[5] Developed jointly at DFKI and UC Davis.
[6] Developed in the Artificial Intelligence Group, University of Bielefeld.

animated agents, independent of a graphics platform, and to synthesize multimodal utterances with prosodic speech, body and hand gesture, or facial expressions [6]. ACE has been used in several conversational agents, e.g., the museum guide MAX and the direction giving information kiosk NUMACK. One hallmark of ACE is its tight coupling of behavior realization, including speech synthesis and animation planning, and behavior execution, which enables smart scheduling and blending/co-articulation of gesture and speech. Assuming that uttering runs incrementally in chunks, animations are formed such that the gesture spans the co-expressive word or sub-phrase within each chunk; between successive chunks, transition movements and silent pauses in speech are adjusted to prepare the synchrony of upcoming behaviors. Originally, the input for ACE was specified in MURML, a representation language addressing the same level of abstraction as BML. ACE is now being made BML-compliant. For BML, this work will result in proposals for new levels of fine-grained, yet player-independent, feature-based gesture description, and for an extension towards complex utterances that consist of multiple "chunks" of coordinated multimodal behavior.

SmartBody[7]**.** *SmartBody* [4] is an open source modular framework for animating embodied characters, based on motion controllers that can be hierarchically interconnected in real-time in order to achieve continuous motion. Controllers can employ arbitrary animation algorithms, such as key frame interpolation, motion capture or procedural animation, as well as schedule or blend other controllers. *SmartBody* was designed around the BML standard and currently supports eight types of BML behaviors: body postures, full or partial body animations, gaze, head motions like nodding and shaking, facial expressions based on action units, speech, event or progress notifications, and the interruptions of prior behaviors. Most behaviors map directly to single motion controllers. SmartBody has been used in a growing number of research projects and training applications, including the SASO-ST virtual human research platform, the ELECT negotiation trainer, the Virtual Rapport studies and the Virtual Patient demonstration.

5.4 Repositories and Tools

The Expressive Gesture Repository[8]**.** In this work, the aim is to help ECAs produce varied gestures from a single representation, based on the agent's expressivity and the relevant semantic dimensions of those gestures. The SAIBA framework supports the inclusion of a BML based gesture repository, either at the behavior planning stage or at behavior realization. An ECA can reproduce gestures from this repository but their application may be limited. Expressivity of behavior is an integral part of the communication process as it can provide information on the current emotional state, mood, and personality of the agents. A set of parameters that affect the qualities of the agent's behavior such as its speed, spatial volume, energy, fluidity have been defined and implemented [3]. During communication, the agent retrieves the gestures definition from the repository and then applies these dynamic variations to the gestures execution.

[7] Developed at the Information Sciences Institute and the Institute for Creative Technologies, University of Southern California.

[8] Developed at IUT de Montreuil, Université de Paris 8.

Moreover, a communicative gesture can have a number of possible meanings, so we have to make sure that a gesture description includes its semantic dimensions. As a first step, we have defined a set of gesture primitives for action gestures that relate to parts of the human body [2], and as a second step, we are currently working on the definition of the relevant dimensions for a family of gestures produced in the space in front of the speaker (gestures with meaning related to: temporal relations, refusal or negation). These descriptions extend possible applications of gestures in the repository, especially when one wants to introduce levels of detail or create variations from a single entry while concerving the meaning of a gesture.

ECAT: The ECA Toolkit[9]. The *ECA Toolkit* (ECAT) aims to allow ECA developers to easily connect any BML behavior-generating system to any behavior realization system. This toolkit assumes that in addition to the three-stage model proposed in the SAIBA framework, the behavior realization stage may be further divided into three distinct processing stages. The first of the three stages is called interpretation, which makes it possible to convert behavior description languages that are not already in BML into well formed BML. The second stage, called compilation, turns BML into low level joint rotations, facial deformations and audio playback directives. Translation, the final stage, involves the rendering of these low level behaviors. Interpreters for different behavior languages and translators for different behavior realizers are being developed.

BCBM Behavior Rule Builder[10]. The Rule Builder is a graphical user interface that allows someone who is not a programmer or an animator to link communicative intent of an animated character to its nonverbal expression of that intent, given a certain context. This linkage is established by way of "rules" authored by the user in a point-and-click manner. The communicative intent is expressed using a preliminary version of FML and the resulting nonverbal behavior is expressed BML. The context can be constructed as an arbitrary XML structure. Both FML and BML are dynamically read into the application from schemas, and therefore the Rule Builder will continue to grow with the development of these standards. The Rule Builder can connect directly to a game engine to provide a real-time preview of resulting behavior. By default behaviors are sent in BML format, but users can specify an XSLT file for translating BML into proprietary animation languages. The Rule Builder has been used with *SmartBody* (using BML) and the Alelo Virtual Culture engine (using Gamebots). The Rule Builder is a part of a behavior generation and authoring toolkit for social training environments that includes a Dialog Builder as described in [12].

6 Conclusion

The SAIBA effort, and BML in particular, is gathering momentum as seen by the ongoing developments and the number of projects that already commit to the

[9] Developed at ArticuLab, Northwestern University.

[10] Developed jointly at University of Southern California and Micro Analysis and Design, currently maintained by CADIA.

framework. Being able to mix and match behavior planners and realizers from a list like the one in the section above and have them work together may be around the corner. However, it is unlikely that the first interoperability attempts will immediately succeed, as integration may reveal unforeseen factors that then need addressing in the framework. It is therefore important to ensure that BML can continue to grow while remaining focused on the interoperability goals. Lessons from integration tests await future publication, but today the prospects for achieving general ECA module interoperability are better than ever, driven by a strong desire to literally build on each other's work.

Acknowledgements. The authors would like to thank those that attended the Vienna workshop as well as those that participated in previous gatherings and online discussions. Particular thanks go to the coordinators of the Vienna and Paris workshops and the EU Network of Excellence HUMAINE (IST-507422) for meeting support. All the individual projects mentioned in this paper are grateful to their respective members for their work and sponsors for their support.

References

1. Cassell, J., Vilhjalmsson, H., Bickmore, T.: BEAT: the Behavior Expression Animation Toolkit. In: Proceedings of ACM SIGGRAPH, Los Angeles, August 12-17, pp. 477–486 (2001)
2. Chafai, N.E., Pelachaud, C., Pelé, D.: A semantic description of gesture in BML. In: Proceedings of AISB'07 Annual Convention Workshop on Language, Speech and Gesture for Expressive Characters, Newcastle, UK (2007)
3. Hartmann, B., Mancini, M., Pelachaud, C.: Design and evaluation of expressive gesture synthesis for embodied conversational agents. In: Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, The Netherlands (2005)
4. Kallmann, M., Marsella, S.: Hierarchical Motion Controllers for Real-Time Autonomous Virtual Humans. In: Panayiotopoulos, T., Gratch, J., Aylett, R., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 253–265. Springer, Heidelberg (2005)
5. Kopp, S., Krenn, B., Marsella, S., Marshall, A., Pelachaud, C., Pirker, H., Thórisson, K., Vilhjalmsson, H.: Towards a Common Framework for Multimodal Generation in ECAs: The Behavior Markup Language. In: Gratch, J., Young, M., Aylett, R., Ballin, D., Olivier, P. (eds.) IVA 2006. LNCS (LNAI), vol. 4133, pp. 205–217. Springer, Heidelberg (2006)
6. Kopp, S., Wachsmuth, I.: Synthesizing Multimodal Utterances for Conversational Agents. In The Journal of Computer Animation and Virtual Worlds 15(1), 39–52 (2004)
7. Lee, J., Marsella, S.: Nonverbal Behavior Generator for Embodied Conversational Agents. In: Gratch, J., Young, M., Aylett, R., Ballin, D., Olivier, P. (eds.) IVA 2006. LNCS (LNAI), vol. 4133, pp. 243–255. Springer, Heidelberg (2006)
8. Neff, M., Kipp, M., Albrecht, I., seidel, H.-P.: Gesture Modeling and Animation Based on a Probabilistic Recreation of Speaker Style. In: Transactions on Graphics, ACM Press, New York
9. Ruttkay, Zs.: Constraint-based facial animation. Int. Journal of Constraints, 6, 85–113 (2001)

10. Ruttkay, Z.s., Zwiers, J., van Welbergen, H., Reidsma, D.: Towards a Reactive Virtual Trainer. In: Gratch, J., Young, M., Aylett, R., Ballin, D., Olivier, P. (eds.) IVA 2006. LNCS (LNAI), vol. 4133, pp. 292–303. Springer, Heidelberg (2006)
11. Thórisson, K.R.: Natural Turn-Taking Needs No Manual: Computational Theory and Model, from Perception to Action. In: Granström, B., House, D., Karlsson, I. (eds.) Multimodality in Language and Speech Systems, pp. 173–207. Kluwer Academic Publishers, The Netherlands (2002)
12. Warwick, W., Vilhjalmsson, H.: Engendering Believable Communicative Behaviors in Synthetic Entities for Tactical Language Training: An Interim Report. In: Proceedings of Behavior Representation in Modeling and Simulation, Universal City, CA (2005)