A Brief History of Function Representation from Gandalf to SAIBA

Hannes Vilhjálmsson and Kristinn R. Thórisson
Center for Analysis and Design of Intelligent Agents
and School of Computer Science
Reykjavík University, Iceland
+354 599 {6323, 6427}
{hannes, thorisson}@ru.is

ABSTRACT
The first half of this paper introduces the aim of SAIBA and the functional markup language (FML) from the perspective of the Center for Analysis and Design of Intelligent Agents (CADIA) at Reykjavik University. The second half provides a brief historic overview of the functional representation of communicative intent in a line of communicative humanoids and related systems, starting with Gandalf and leading up to one of the early proposals for FML in the SAIBA framework.

Categories and Subject Descriptors
I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *frames and scripts, representation languages, representations*.

General Terms
Algorithms, Design, Standardization, Languages, Theory.

Keywords
Embodied Conversational Agents, Functional Representation, Multimodal Communication, Human Computer Interaction.

1. INTRODUCTION
As the SAIBA consortium gears up for the second phase of the planned work on representations for multimodal generation, we would like to summarize what we consider some of the key aspects of this work as well as give a view of some of the historical roots that the effort grew out of.

The SAIBA (situation, agent, intention, behavior, animation) effort exists first and foremost for the purpose of increasing the synergy within the research community focused on multimodal communication in robots and virtual humanoids. Computer graphics work has enjoyed enormous success in standardization efforts for what we see as the lowest layer in a stack upon which systems and abstractions layer ever-increasing complexity and, eventually, intelligence. The observation is simple: As more people do research in interactive humanoids the potential for duplication of effort increases. The consortium has addressed the problem of multimodal generation "one abstraction level" above the graphics level; this resulted in the BML effort, which was based on a lot of prior work.

At the BML level we see groupings of primitive moves, what in robotics is called e-moves, into larger, more complex sets of instructions, into what in the robotics world is often referred to as action and which in BML are called behaviors. BML is thus a language for describing events that are supposed to happen. The events are not as simple as graphics commands (otherwise they would not save the developer any time), so they cannot represent the same fine level of detail – and that is precisely the point. They are higher level than e-moves and therefore can be used to program large sets of e-moves in single strokes. However, human behavior is highly complex, and just as basic computer graphics commands are not well-suited to describe complex humanoid behaviors, BML is not convenient for representing long chains of multimodal events – what in the A.I. world are called plans.

Enter FML – functional markup language. The aim with FML is to develop the next level of description language up from BML, one that can describe what should happen in a multimodal agent at what we could call a functional level – representing in essence what the agent's behavior(s) should achieve – it's goals.

The SAIBA consortium's approach to this effort is based on one tenet that is extremely important for the success of achieving the stated aims, i.e. increased collaboration, ease of sharing results and actual working systems: a clear separation of representation language and the processes that produce and consume the language. This is, for the most part, practically motivated and is based on a long history and can, in our view, help keep the effort on a prosperous track. A successful separation keeps open the possibility that anyone can create their own planning mechanisms. For this to be possible the future FML cannot and must not put constraints on the kinds of processes that consume and produce it. Whether this is possible remains to be seen. But since the main research focus in artificial intelligence and communicative humanoids on the topic of multimodal generation are the mechanisms that control and produce the behaviors, this must be a free variable, unconstrained by the languages that the processes work with. The language, FML, will describe the intentions that an agent may have in what it does. This, of course, should be possible to do without saying anything about the mechanisms that are required to manipulate those intensions.

It is important to follow the basic idea behind the SAIBA effort of layers or bands – that is, a given markup language is not only limited in that there are details – lower-level things – that it cannot (and should not) represent, there will also be larger – higher level – things that it should not represent; BML and FML are constrained to bands of operation. These bands are limited by time and scale, that is, the timescales covered by BML are smaller than those covered by FML. Likewise, as we build FML there

may be large timescales for which FML will be inappropriate; this, however, remains to be seen.

We will now turn to some of the historical precedents for the current FML efforts.

2. FUNCTION / INTENTION REPRESENTATION IN GANDALF

In the Ymir architecture, on which the communicative humanoid Gandalf was built [1] (Figure 1), a number of ideas were presented that relate to the present effort. The main components relevant to FML include the *Action Scheduler* (AS), which could receive and execute goals representing both functional and behavioral specifications.

Figure 1: The Gandalf/Ymir was capable of real-time face-to-face conversations with human users. Notice Gandalf's gaze responding to the hand gesture's interpreted function - i.e. pointing - within (a human- like delay) of 300-400 ms.

On the perception side, Ymir had a set of processes called *Multimodal Descriptors* that could aggregate information from *Unimodal Perceptors* (both these modules worked with real-time perceptual data generated by the behavior of a person); the output of these were "sketches" – descriptions of human behavior – at various levels of detail. Examples of some of the higher-level descriptors are given in Table 1.

Table 1. Example higher-level descriptors in Gandalf

giving-turn
taking-turn
wanting-turn
want-back-ch-feedb
has-turn
addressing-me
greet
greet-happily

Upon the reliable detection of any of these descriptors (and their temporal inter-relationships), *Decider* modules would fire goals for being achieved through movement or speech in the Gandalf agent. The Action Scheduler would receive these goals and generate the appropriate animation commands, (i.e. e-moves). In Ymir the AS is the last stop before ballistic execution of animation. To enable interruptability, the ability to cancel actions quickly for any degree of freedom, the AS would never commit more than 200 ms at a time to the animation level below. The goals generated by the Deciders could specify the shape/look of an action, e.g. *hand-raise-palm-forward*, what it should achieve, e.g. *greet*, or both, e.g. *greet-happily*. The AS resolved these goals down to the graphics level by selecting between options such as whether to greet with wink, nodding or waiving, etc., down to the level of primitive animation commands. Thus, the AS handled, in one place with a single mechanism, both what we would later refer to as the BML level and FML level.

3. FRAMES OF FUNCTIONS IN REA

The general approach of Gandalf/Ymir was maintained in a later agent called REA (Figure 2), although REA's architecture replaced shared descriptor blackboards with a fixed messaging pipeline that passed around a multimodal *Frame* [2].

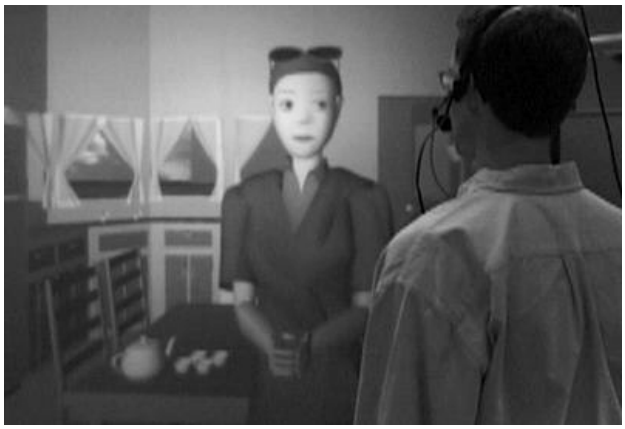

Figure 2. REA was a real-estate agent capable of multimodal natural language generation and understanding.

A user event generated an input Frame in REA's system that contained a field for observed visual or audible behavior and two fields for functional interpretations of these behaviors: A *Propositional* interpretation (content related) and an *Interactional* (process related) interpretation. These interpretations were added by an *Understanding Module* before a *Decision Module* would then use the interpretations to create an appropriate response for REA. The response was encoded in an output Frame similar to the input Frame in that it contained the same Propositional and Interactional fields, which were now filled with communicative functions that needed to be realized by the agent. The output Frame was sent through a *Generation Module* that generated appropriate behaviors, fulfilling the functions by placing a description of those behaviors in a special output field.

Therefore REA's central decision mechanism only operated on a functional representation of the user's input and produced only a functional representation of her communicative intent.

REA's Interactional functions were in part drawn from Gandalf's descriptors and are shown in Table 2.

Table 2. Interactional functions in REA

giving-turn
taking-turn
wanting-turn
keeping-turn
dropping-turn
listening *(to a speaker)*
wanting-feedback
giving-feedback
expecting *(some input)*
present *(within conversational distance)*
inviting *(to start a conversation)*
leaving*(the conversation)*

REA's Propositional functions were in the form of *Speech Acts*, which were for the most part domain dependent, but divided into *imperatives*, *interrogatives* and *declaratives*. There was also a special *ritual* category that contained a *greeting* and a *farewell*.

4. FML IN SPARK

Influenced by work on REA and the need for something more lightweight, BEAT was built as a tool for generating multimodal co-verbal behavior based on analyzing the text to be spoken [3]. Unlike Gandalf and REA, BEAT only dealt with multimodal generation, not perception. The most comprehensive implementation of BEAT existed as part of the Spark system (Figure 3), which automated avatar behavior in an online virtual environment based on chat messages exchanged by its users [4].

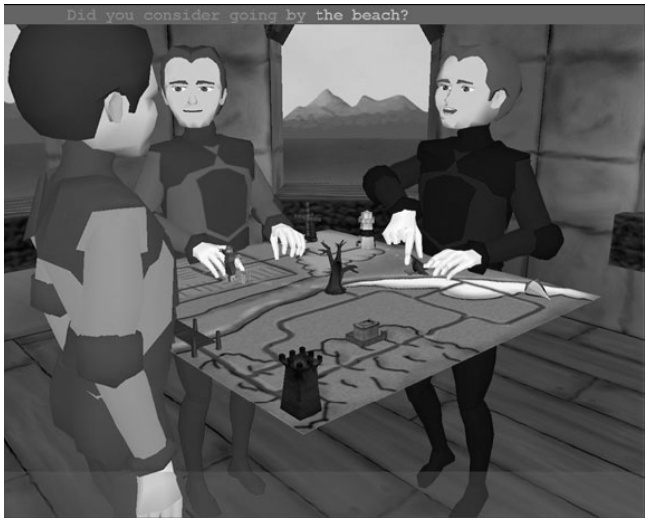

Figure 3. Online avatars automated in Spark based on functional annotation of text exchanged by users.

In this implementation, each chat message got analyzed and annotated in terms of various discourse functions (Figure 4).

```
<utterance scene="map1" speaker="person1"><clause>
 <theme><turn type="take">
  <action><new> give </new></action>
  <object> him </object></turn></theme>
 <rheme>
  <turn type="give" target="person2">
  <emphasis type="phrase">
   <reference type="visual" target="map:mine">
    <reference type="textual" source="person3">
     <object id="map:mine"> some
      <emphasis type="word">
       <new> gold </new>
      </emphasis>
     </object>
    </reference></reference></emphasis></turn></rheme>
</clause></utterance>
```

Figure 4. The text "give him some gold" automatically annotated in terms of discourse function in Spark .

The discourse functions annotated by the improved BEAT of the Spark system are shown in Table 3.

Table 3. Discourse functions in the Spark FML

topic-shift
turn *(types: take, keep or give)*
clause *(types: normal, exclamation or question)*
new *(lexical givenness)*
theme *(link to previous utterance)*
rheme *(new contribution to conversation)*
emphasis
contrast
reference *(types: visual or textual)*
illustrate *(elaborate feature through illustration)*
grounding *(types: request)*

These functions would then be mapped into supporting nonverbal behavior on the receiving end (according to existing empirical data on face-to-face discourse) for a full multimodal delivery.

The functions were drawn from the literature on discourse and conversation analysis, and represent some of the most common elements that give rise to conversational nonverbal behavior. Some of them are *Interactional* in nature (such as turn) while others are *Propositional* (such as contrast). However, some are difficult to classify according to these categories, such as the process of grounding, which may rely on both.

In BEAT the function annotations were done using XML tags placed directly within the annotated text. The term *Function Markup Language* (FML) was used to describe these tags in the Spark system to contrast them with the set of tags used to describe the supporting *Behavior* (BML). This naming of the two different tag sets has been maintained in the SAIBA framework, but the actual tag structure has evolved.

5. THE EVOLVING FML IN SAIBA

Perhaps one of the largest differences between BEAT tags and SAIBA tags is that the latter breaks free of the strict hierarchical ordering of tags with the introduction of *Synch Points* [5]. SAIBA XML descriptions can be flat, with ordering constraints provided through Synch attributes. This allows partially overlapping tags.

While BEAT could make the assumption that the input would be the text to be spoken, and therefore that FML tags could simply be placed around appropriate text elements, the SAIBA FML representation can make no such assumption. The generation of the text itself may not have occurred at the time of function description.

That is why the FML break-out group at the Reykjavik SAIBA workshop in 2005 proposed to divide FML tags into two sets. The first set defines certain basic functional or semantic *Units* that are associated with the communicative event. The second set includes *Operations* that essentially operate on previously defined units to apply certain functional effects on them. The initially proposed units are listed in Table 4.

Table 4. First SAIBA FML proposal: Units

participant
turn
topic
performative *(speech act)*
content *(detailed proposition)*

These units are ordered here from the widest *scope* to the smallest *scope*. That means that the widest scoped FML may contain one or more of the smaller scoped elements. The Operation tags can affect any of these units (and therefore their scope will vary greatly). A preliminary list of the suggested Operation tags is provided in Table 5.

Table 5. First SAIBA FML proposal: Operations

emphasis
contrast
illustration
affect
social *(relational goals)*
cognitive *(meta-cognitive e.g. difficulty of processing)*
certainty *(producer's certainty of unit's truth)*

An example of how an FML block could be constructed using these two sets of tags was given in [6]. The example is reproduced here in Figure 5. As the SAIBA effort focused more on BML during the following phase, this early FML proposal has not been fleshed out so far, and is therefore very much work in progress

```
<!- Defining units – first set ->
<participant id="ali" role="speaker"/>
<participant id="trainee" role="addressee"/>
<turn id="turn1" start="take" end="give">
    <topic id="topic1" type="new">
        <performative id="perform1" type="enquiry">
            <content>goal trainee ? here</content>
        </performative>
    </topic>
</turn>
<!- Operating on units – second set ->
<emphasis type="new">perform1:here</emphasis>
<affect type="fear">perform1:goal</social>
<social type="maintain_distance">trainee</social>
```

Figure 5. An example of an FML description that might result in leaning away and speaking "What are you doing here?"

6. CONCLUDING REMARKS

Looking back over these related projects, one thing is striking: The first systems focused on the essential mechanism or process of maintaining real-time dialogue, while the later ones start looking into the presentation of content. There seems to be a relatively good agreement about the process or interactional functions, so perhaps this is a good place to start with a shared specification. It is important that interactional functions continue to be first-class citizens in any FML representation and that they not only be useful for generation of behavior but also for interpretation of behavior.

7. REFERENCES

[1] Thórisson, K. R., 1996. Communicative Humanoids: A Computational Model of Psychosocial Dialogue Skills. Ph.D. Thesis, Massachusetts Institute of Technology, MA.

[2] Cassell, J., Vilhjálmsson, H., Chang, K., Bickmore, T., Campbell, L. and Yan, H., 1999. Requirements for an Architecture for Embodied Conversational Characters. In Computer Animation and Simulation '99 (Eurographics Series). Vienna, Austria: Springer Verlag.

[3] Cassell, J., Vilhjálmsson, H., and Bickmore, 2001. BEAT: the Behavior Expression Animation Toolkit. In Proceedings of ACM SIGGRAPH, Los Angeles, Aug. 12-17, p.477-486.

[4] Vilhjálmsson, H., 2005. Augmenting Online Conversation through Automated Discourse Tagging. In Proceedings of The 6th Annual Minitrack on Persistent Conversation at the 38th Hawaii International Conference on System Sciences, Jan. 3-6, 2005, Hilton Waikoloa Village, Big Island, Hawaii, IEEE, 2005

[5] Kopp, S., Krenn, B., Marsella, S., Marshall, A., Pelachaud, C., Pirker, H., Thórisson, K., Vilhjálmsson, H., 2006. Towards a Common Framework for Multimodal Generation in ECAs: The Behavior Markup Language. In Proceedings of the 6th International Conference on Intelligent Virtual Agents, Aug. 21-23, Marina del Rey, CA

[6] Vilhjálmsson, H. and Marsella, S., 2005. Social Performance Framework. In Proceedings of the Workshop on Modular Construction of Human-Like Intelligence at the 20th National AAAI Conference on Artificial Intelligence, July 9th, Pittsburgh, PA