MultiML – A General Purpose Representation Language for Multimodal Human Utterances

Manuel Giuliani
Robotics and Embedded Systems Group
Department of Informatics
Technische Universität München
Boltzmannstraße 3
D-85748 Garching bei München, Germany
giuliani@in.tum.de

Alois Knoll
Robotics and Embedded Systems Group
Department of Informatics
Technische Universität München
Boltzmannstraße 3
D-85748 Garching bei München, Germany
knoll@in.tum.de

ABSTRACT

We present MultiML, a markup language for the annotation of multimodal human utterances. MultiML is able to represent input from several modalities, as well as the relationships between these modalities. Since MultiML separates general parts of representation from more context-specific aspects, it can easily be adapted for use in a wide range of contexts. This paper demonstrates how speech and gestures are described with MultiML, showing the principles—including hierarchy and underspecification—that ensure the quality and extensibility of MultiML. As a proof of concept, we show how MultiML is used to annotate a sample human-robot interaction in the domain of a multimodal joint-action scenario.

Categories and Subject Descriptors

H.5 [**Information Systems Applications**]: Information Interfaces and Presentation—*General*

General Terms

Languages, Standardization

1. INTRODUCTION

Humans communicate with one another in many ways. They use their whole body to give information about their current status, their emotions, and their intentions. But how can the information from different modalities—speech, gestures, gazes, etc.—be described in a way so that autonomous agents, including computers and robots, are able to understand these human utterances? Hitherto, there is no single data format that is intended to represent input from various modalities and for different contexts. That is because of the following reasons:

- *Area specialisation.* Most researchers develop data formats with the needs for their scientific area in mind. Therefore, a linguist only focuses on representations for speech, a gesture researcher provides a representation for gestures and so on.

- *Focus on single domain.* Data formats for unimodal or multimodal representation often emerge from research projects that focus on a certain domain or on a special aspect of a single modality. Hence, these formats are usually tailored to the corresponding project and are hard to adapt to other domains.

- *Unavailability of data.* With the possible exception of speech, rarely any modality is really understood well enough so that it can be described in an appropriate way. In addition, the relations between the single modalities and the integrated meanings they build together, are even less well explored.

In this paper, we present MultiML, the Multimodal Markup Language. We developed MultiML to represent multimodal human utterances in different contexts. Of course, this language must deal with the issues we described above. However, research results of recent years regarding the syntax and semantics of modalities like gestures or gazes, have led to a situation where we can at least partially understand the relations between the different modalities in natural dialogue. Also, advances in linguistics, computer science, and robotics, allow us to demonstrate our findings on a physically available platform for human-robot interaction. We will show that MultiML meets the needs of a researcher who wants to describe his/her observations of human dialogues, as well as the needs of the developer of a computer or robot which must understand it's human counterpart in a dialogue situation that involves more than one modality. For the beginning, we will start with the representation of speech and gestures and the relations between these two modalities.

The remainder of this document is organised as follows: Section 2 reviews existing formats for representation of multimodal content. Section 3 gives an introduction to the JAST project, for which MultiML was developed initially. After that, Section 4 presents the features of MultiML and shows how speech and gestures are represented. In Section 5, we demonstrate MultiML's unique property to be easy to adapt to a given context, by fitting it to the JAST domain in a few simple steps. Finally, Section 6 concludes this publication and gives an outlook for future perspectives on MultiML.

ICMI'08, October 20–22, 2008, Chania, Crete, Greece.

2. RELATED WORK

In recent years, some data formats for representation of multimodal data have been proposed. These formats can roughly be separated into those which represent the syntactic relations of modalities and others that provide a semantic representation of the content. While the first are often used in output generation for virtual agents, the latter are mostly intended to be the input to decision making components in multimodal systems. Nearly all of these formats have in common that they have been developed for the use of one domain only. In the following we will review the multimodal representation languages that are close to what we want to achieve with MultiML.

Kranstedt et al. [1] present MURML, the Multimodal Utterance Representation Markup Language, which is used to describe speech and gestures that are interpreted by a virtual agent. MURML's focus is on the exact description of body limbs, body movements, and the exact representation of timing events, so that the virtual agent executes the right movements with the correct body parts or combination of speech and gestures at each time point. An interesting aspect of MURML is that it utilises names for hand shapes, hand orientation, and locations that are taken from HamNoSys [2], which is a representation formalism for sign languages, based on the form of the gestures.

Landragin et al. [3] introduced MMIL, the MultiModal Interface Language. MMIL can be used to specify the multimodal communication between the components of a multimodal system. MMIL is organised hierarchically and differentiates between several layers of dialogue information: phones, words, phrases, and utterances. This information is organised in a set of entities, with each entity standing for an event that happens at a defined time point. The second main class in MMIL are the so-called participants that name the parts of a multimodal system, which communicate with each other. Landragin et al. demonstrate, how MMIL can be transferred from its original domain to a new one. However, porting MMIL to a new domain requires changing parts of the format completely.

Gibbon et al. [4] present CoGesT, the Conversational Gesture Transcription. CoGesT is a human and machine readable format that is intended to be used for annotation of gestures in multimodal corpora and for automatic recognition of gestures in conversations. CoGesT describes gestures with source and target location, body part, body shape, and trajectory between the source and target location. This information is presented in feature vectors which are context-independent, because the locations in the vector are given relatively to the body of the person who is uttering the gesture. Unfortunately, it is not possible to extend CoGesT with context-dependent information, and the labels for locations, hand shapes, etc. are non-intuitive abbreviations: e.g. the abbreviation "15m" stands for the location "on the lap".

Chai [5] shows an approach for a semantic representation of multimodal input. This representation can be used for single modalities as well as for an integrated representation of several modalities. It consists of two parts: the first part depicts the intention behind an utterance, while the second part represents the utterance's attention; i.e. it lists objects in the environment that are linked to the utterance. The representation can be applied to various modalities, in Chai's case to speech and touch pen input, so that the multimodal fusion of data represented in this format can be done by unification. Chai also shows how the system can be used to resolve ambiguities in dialogue discourse.

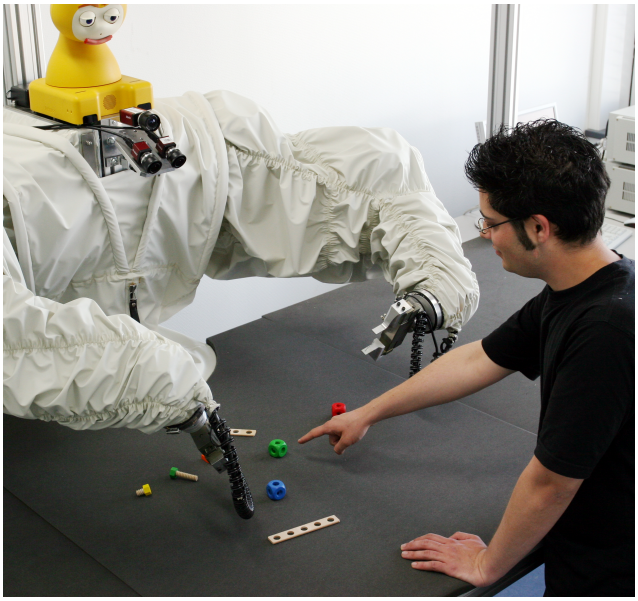

Figure 1: Example for multimodal human robot interaction: the human says "*Take this cube.*" and points at the green cube.

3. JAST

MultiML was developed in the context of the European JAST project. The acronym JAST stands for Joint-Action Science and Technology. The two main goals of JAST are to investigate the cognitive, neural, and communicative aspects of jointly-acting agents, both human and artificial, and to build jointly-acting autonomous systems that communicate and work intelligently on mutual tasks. The research results from the JAST project are amongst others implemented on a robot [6] that consists of two industrial robot manipulators mounted to resemble human arms and an animatronic head so that a human can interact with the system across a table. Various cameras are installed for the recognition of objects on the table [7], for gesture recognition [8], and for tracking the human. Force/torque sensors and grippers with position sensors on each arm complete the installation. Input received on all of these channels is continuously published through the Internet Communications Engine middleware (Ice) [9]. Messages from the various input channels are processed by a multimodal fusion component [10], which parses the text and combines it with non-verbal information to produce combined hypotheses representing the user requests. The fusion hypotheses are then sent to the dialogue manager, which selects an appropriate response. Figure 1 shows the robot in interaction with a human. For a more detailed description of the JAST robot please refer to [11].

4. MULTIML

This section describes MultiML in more detail. After an introduction that shows an example which motivates MultiML's properties (Section 4.1), we will explain how speech

(Section 4.2) and gestures (Section 4.3) are represented in MultiML. Finally, we show how links between speech and gestures are represented (Section 4.4).

4.1 Motivation and Properties

In the JAST project, we are currently working towards a scenario in which human and robot build a railway sign together. The railway sign is assembled of a wooden toy construction set called Baufix. To assemble the railway sign, two cubes, three slats, and three bolts are needed. Figure 2 shows a schematic representation of the sign. The scenario consists of two phases: in the first phase the system instructs the human how to build the railway sign, which should show the robot's communicative abilities. In the second step, the human and the robot build the railway sign a second time. Assuming that the human learned how to build the railway sign, the robot can show its non-verbal communication skills, which also include anticipation of the human's actions and error detection.

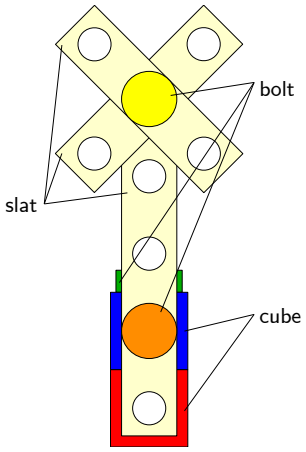

Figure 2: JAST scenario: human and robot build together a railway signal of a wooden toy construction set called Baufix. The railway signal consists of two cubes, three slats, and three bolts.

The scenario was chosen for several reasons: first of all the scenario is perfect for showing the system abilities of the JAST robot. In addition, the scenario is well suited to show joint-action of human and robot on a small scale. The concepts and ideas behind this scenario can be extended to more complex scenarios and industrial settings. We chose an example from a JAST human robot dialogue to illustrate the use of MultiML for human robot interaction, which will demonstrate all the language's features. The example, depicted in Figure 1, shows one of the basic combinations of two modalities in joint-action: the simultaneous use of speech and gestures for deictic references.

In the example, the human who works together with the JAST robot wants the system to pick up a green toy cube. Thus, the human points at the cube and says "*Take this cube.*". This seemingly simple example of multimodal interaction already contains all of the aspects of MultiML, which we want to discuss in this publication. For the representation of speech we need to generate a logical form that depicts the syntactic information of the spoken sentence as well as information about the actions that have been expressed by saying the sentence and the entities (i.e. agents and objects) that were mentioned. For the gesture channel we want to have a representation of the observed gesture posture and the body part used to execute the gesture, of the locations where the gesture took place, and of the timing relations among the single parts that build the whole gesture. Of course, we also want representations for information of any automatic classifier that is used for the recognition of speech and gestures, for example information for multiple hypotheses and their confidence values. However, before we start explaining the details of MultiML, we need to discuss the principles on which MultiML is build to ensure the quality of the markup language.

The main goals for MultiML are *generality* and *extensibility*. That means that our approach should be *general* enough to be applicable in various contexts and different application scenarios. Supporting generality also requires a data format with a scalable level complexity. In other words, the data format always should be just as complex as needed for any given context. The second goal is to develop a data format that is *extensible*, which has several motivations: first of all, we believe that it is impossible to develop a data format that is able to represent all human utterances from the beginning. Therefore, it is better to start with the representation of selected modalities and contexts, keeping in mind to extend the data format at later stages. Second, it is our desire to keep MultiML open to other ways to represent human utterances: for example, if someone develops a novel format to represent human emotions, it should be possible to import this format into MultiML.

The next important goal of MultiML is its *context independence*. By that, we mean that MultiML should be used to represent a given context, but it should not be tied to this context. To reach this goal, we believe that it is necessary to determine the context-independent and context-dependent parts of any modality represented in MultiML. These two parts have to be separated, so that the context-dependent parts can be generated quickly or even automatically, when MultiML is adapted to a new domain. To illustrate this, we will explain in Section 4.3 which parts of a gesture are context-independent or context-dependent and how these parts are separated in MultiML

Another property of MultiML is its *hierarchical structure*. This feature is needed to keep MultiML's granularity adjustable. That means that the format displays only the information that is needed at a given time. For example, applications that want to control a virtual agent need much more precise information about hand shapes, movement trajectories, etc., than a dialogue system that is interested in interpreting the input of an automatic gesture recogniser. Hierarchy can be found at several levels in MultiML: the utterances in the single modalities are organised hierarchically, for example gestures consist of several phases, as we will show later, and these phases can be described in different levels of complexity. Another hierarchy can be found in the timing relations among the different utterances. This hierarchy is used to display how utterances from different modalities are nested into, precede, or succeed each other, much like in the temporal logic by Allen [12].

One other important property of MultiML is based on the concept of *underspecification*. Underspecification occurs when features of a representation are omitted because they are dispensable for the current context. That means that

the representation only shows the necessary information and this way becomes better readable and easier to compute. Underspecification can be achieved directly by omitting unnecessary parts of the representation, or indirectly by using templates that stand for complex correlations. For example, a template "pointing gesture" could describe a hand posture, where the hand is hold in a way that only the index finger is stretched, while the remaining fingers are fastened. This makes it easier to annotate visual modalities, but leaves out unnecessary information such as the exact position of each finger. Underspecification also ensures that MultiML can already be used in an early stage of development.

Finally, we respected *computational aspects* during the development of MultiML. This means that we wanted to ensure that MultiML cannot only be processed by a computer, but also that the facilities that are needed for automatic recognition of modalities are present. For example, timestamps in MultiML are written down in milliseconds since 01.01.1970, which is the usual measure for timestamps in many popular programming languages, including C++ and Java.

4.2 Representation of Speech

In the JAST project, we are using Combinatory Categorial Grammar (CCG) to parse and represent speech input that comes from an automatic speech recogniser. CCG was introduced by Ades [13] and Steedman [14]. It is an extension to the Categorial Grammar (CG) of Ajdukiewicz [15] and Bar-Hillel [16]. Traditional context-free grammar formalisms use a top-down approach for parsing sentences, while combinatory grammars utilise a bottom-up approach, which brings advantages in computability and grammar development. Due to the addition of combinatory logic to the grammar formalism, CCGs produce a semantic representation of a sentence during the parsing process. JAST uses a CCG that was implemented with OpenCCG. OpenCCG [17] is a Java-based implementation of the CCG formalism. It is capable of both, parsing and realising sentences; that means it can translate utterances into a logical form as well as take a given logical form and convert it back to a sentence. OpenCCG generates hybrid logic expressions for the parsed sentence instead of combinatory logic, as explained in [18]. Figure 1 shows such a hybrid logic formula that was parsed with the JAST grammar and represents the sentence from the multimodal example "*Take this cube.*".

$$
\begin{aligned}
@_{t1-action}(&\text{take-verb} \wedge \\
&\langle \textsc{mood} \rangle \text{ imp} \wedge \\
&\langle \textsc{Actor} \rangle\ x1 - animate - being \wedge \\
&\langle \textsc{Patient} \rangle\ (\ c1 - thing \wedge \text{cube-np} \wedge \\
&\qquad \langle \textsc{det} \rangle \text{ dem-prox} \wedge \\
&\qquad \langle \textsc{num} \rangle \text{ sg }))
\end{aligned}
$$

Figure 3: Hybrid logic formula that was generated with the JAST combinatory categorial grammar for the sentence "*Take this cube.*". The formula shows the syntactic aspects of the sentence and names all actions and entities in the sentence with identifiers.

To understand this logic formula, we have to illustrate the two concepts of *nominals* and *diamond operators* that are utilised in hybrid logics. *Nominals* can be seen as identifiers that are used to name parts of the logical form. In the present case we used nominals to name the actions expressed in a sentence and the entities that are involved in the action. In the example, the nominal *t1-action* is used to name the take action expressed in the sentence, while the two nominals *x1-animate-being* and *c1-thing* name the actor that should execute the requested action and the cube that is involved in the action, respectively. We can also see the *diamond operators* ⟨MOOD⟩, ⟨ACTOR⟩, ⟨PATIENT⟩, ⟨DET⟩, and ⟨NUM⟩. These operators represent the syntactic properties of the parsed sentence, including such information as that the sentence was uttered in imperative mood or that a proximal demonstrative was used as determiner to further specify a certain cube. For a more detailed description of the use of CCG, OpenCCG, and hybrid logic in the JAST project, please refer to [10].

For MultiML we translate hybrid logic formulas into an XML representation and enrich it with additional information. Figure 4 shows the MultiML representation of the hybrid logic formula in Figure 3. The formula itself is denoted from lines 4–13. The nominals are denoted in the *id* attributes, while the diamond operators are either written down as relations, nested in the *rel* tags, or additional attributes to the *node* tags. We enriched the logic formula by the *target* tag (line 3), which contains the sentence that was parsed. Target and logic form are nested into *speech* tags (lines 2 and 14) to show that the utterance belongs to the speech modality. The whole expression is wrapped in *utterance* tags (lines 1 and 15) that are the general tags to represent utterances in MultiML and are also used for other modalities, as we will show in Section 4.3. The utterance tags are extended by the attributes *id*, *startTime*, *endTime*, and *confidence*, which will also be further explained in the next section.

```
1 <utterance id="s1" startTime="1000"
      endTime="5000" confidence="100.00">
2  <speech>
3   <target>take this cube .</target>
4   <lf>
5    <node id="t1-action" pred="take-verb" mood="
         imp">
6     <rel name="Actor">
7      <node idref="x1-animate-being"/>
8     </rel>
9     <rel name="Patient">
10     <node id="c1-thing" pred="cube-np" det="
           dem-prox" num="sg"/>
11    </rel>
12   </node>
13  </lf>
14 </speech>
15 </utterance>
```

Figure 4: Representation of speech in MultiML. The example sentence in this utterance is "*Take this cube.*". The representation is based on hybrid logic that was generated by the combinatory categorial grammar implementation OpenCCG.

4.3 Representation of Gestures

Before we describe our approach for the representation of gestures in full detail, we have to discuss the types of gestures we aim to represent with MultiML. McNeill [19] introduced "Kendon's continuum", named after Adam Kendon, in which he classifies gestures into several following subclasses: *spontaneous gesticulation*, *language-slotted gestures*, *panto-*

mime gestures, *emblems*, and *signs*. We are interested in the kind of gestures that either accompany speech (i.e. spontaneous gesticulation) or replaces it (i.e. language-slotted gestures). Pantomime gestures and emblems, which are culturally established morphemes such as the "thumbs up" sign can also occur without speech and are easy to represent. Sign languages are full languages by themselves with an own grammar and structure and there exist representation formalisms for sign languages, for example the already mentioned HamNoSys.

The gesture representation of MultiML closely follows the work of Adam Kendon. Kendon [20] separates single gestures into so-called *gesture units*. Each gesture unit consists of three phases: *preparation*, *nucleus*, and *recovery*. During *preparation* the person expressing a gesture moves the body part that is used to execute the gesture from a resting position to the position where the gesture should be expressed. The part of the gesture that most people consider as the part carrying the information of the gesture is called the *nucleus*, which consists of the *stroke* and an optional *post-stroke hold*. The stroke takes place at a defined location and is usually expressed by a short, fast movement. For example, in a pointing gesture the index finger of a hand is extended to point somewhere and then taken back to the original position. Sometimes the body part that was used to express the gesture can be held still for some time, which is called *post-stroke hold*. After the nucleus, the hand is moved again to a resting position in the *recovery* phase. Kendon breaks down this simple gesture model into a more sophisticated model. He combines preparation phase and nucleus into the so-called *gesture phrase*. Each gesture unit can contain one or more gesture phrases, but only one recovery phase. Just imagine a situation where a human says "*Move the cube from here to there.*", accompanied by a gesture utterance that involves two pointing gestures. This gesture unit contains two gesture phrases, in which the person moves the hand from one point to the other in preparation to execute the pointing gesture in a stroke two times. After these two gesture phrases, the hand is moved back to a relaxed position in the recovery phase.

The gesture unit structure by Kendon builds the base for the gesture representation in MultiML, as shown in Figure 5. The three phases of a gesture—preparation, nucleus with stroke and hold phase, and recovery—are translated into an XML representation. We also defined an XML schema definition for MultiML, which allows to establish constraints on the format, for example that a gesture unit can contain several gesture phrases, but only one recovery phase. This schema will be further specified in Section 5.

```
1 <gesture>
2   <gesturePhrase>
3     <preparation />
4     <nucleus>
5       <stroke />
6       <hold />
7     </nucleus>
8   </gesturePhrase>
9   <recovery />
10 </gesture>
```

Figure 5: Kendon's *gesture units* build the base for the representation of gestures in MultiML.

Similar to the speech utterances, gesture units are also nested into utterance tags to form the utterance hierarchy. We extended this base representation by several features that meet the requirements for gesture representation that we stated earlier. First, we want to display the timing relations of the single phases of a gesture. Figure 6 shows how timings are represented in MultiML. We added the attributes *startTime* and *endTime* to the *utterance* tags and to the tags for *preparation*, *stroke*, *hold*, and *recovery* (lines 4, 6, 7, and 10). The time points are given in "milliseconds since 01.01.1970", which allows an easy implementation of MultiML on a computer afterwards. This figure also shows how utterances are nested inside the utterance tag in MultiML to follow the hierarchy principal.

```
1 <utterance startTime="2000" endTime="4000">
2  <gesture>
3   <gesturePhrase>
4    <preparation startTime="2000"
         endTime="2200" />
5    <nucleus>
6     <stroke startTime="2200" endTime="2500" />
7     <hold startTime="2500" endTime="3000" />
8    </nucleus>
9   </gesturePhrase>
10  <recovery startTime="3000" endTime="4000" />
11 </gesture>
12 </utterance>
```

Figure 6: The base representation for gestures is extended by the attributes *startTime* and *endTime* to represent the timing relations of the three phases of a gesture unit.

The second extension to the basic representation of gestures is the addition of context-dependent content. Figure 7 shows how these contents are included in the representation. The context-dependent content describes the *body part* (line 3) that is used to execute a gesture, the *locations* (lines 6–11, 15–17, and 21–26), where for example the hand is placed during the gesture phases, and the *posture* (line 13) that is made to express the gesture in the stroke. Locations can be either described by vague descriptions like "tableMiddle" or by providing coordinates that meet the requirements of a given domain. These parts are not context-dependent themselves, but their content is. By using XML and XML schema to describe the format, the implementation of MultiML allows us to keep the content of the context-dependent parts of a gesture separate from the remaining gesture description, so that these contents can be generated automatically from descriptions given by a developer who adapts MultiML to his/her domain.

The third addition to the basic representation of gestures is shown in Figure 8. This partial representation of the gesture shows on the one hand, how added identifiers, similar to the hybrid logic nominals described in Section 4.2, can be used to name and refer to whole gestures (line 1) or, even more finely, gesture strokes (line 5). This will be useful for the linkage of speech and gesture, as we will demonstrate in Section 4.4. On the other hand, Figure 8 also shows how confidence values, which can be derived from automatic gesture classification systems, can be added to a whole gesture (line 1) or the stroke phase of a gesture (line 5). This way,

```
1  <utterance startTime="2000" endTime="4000">
2   <gesture>
3    <bodyPart>handRight</bodyPart>
4    <gesturePhrase>
5     <preparation startTime="2000"
          endTime="2200" />
6     <startLocation>
7      <locationDescription>tableLeft
           </locationDescription>
8     </startLocation>
9     <endLocation>
10     <locationDescription>tableMiddle
           </locationDescription>
11    </endLocation>
12    <nucleus>
13     <stroke posture="pointing" startTime="2200"
           endTime="2500" />
14     <hold startTime="2500" endTime="3000" />
15     <location>
16      <locationDescription>tableMiddle
           </locationDescription>
17     </location>
18    </nucleus>
19   </gesturePhrase>
20   <recovery startTime="3000" endTime="4000">
21    <startLocation>
22     <locationDescription>tableMiddle
           </locationDescription>
23    </startLocation>
24    <endLocation>
25     <locationDescription>tableLeft
           </locationDescription>
26    </endLocation>
27   </recovery>
28  </gesture>
29 </utterance>
```

Figure 7: Representation of context-dependent content in gestures, including posture types and locations.

multiple hypotheses for a gesture can also be represented in MultiML.

This section explained the basic way to represent gestures using Kendon's gesture units. Building on this basis, we demonstrated how timing relations, context-dependent content, identifiers and confidence values can be added to MultiML. In the following section we will describe how the representations for speech and gestures can be linked in between each other.

4.4 Relation of Speech and Gesture

In the former sections we introduced nominals and identifiers in the representations of speech and gestures respectively. These are used to name whole utterances or parts of them and refer to them later. This functionality can now be used to establish links between modalities. Figure 9 shows parts of the speech and gesture examples that were shown in Figures 4 and 7. We extend these representations by the introduction of a new attribute *linkedTo*, which can be placed in any XML tag that contains the *id* attribute. This way the relation between the sentence "*Take this cube.*" with id *s1* (line 1) and the pointing gesture with id *g1* (line 11) can be represented. One can also link the specific gesture stroke with id *s1-stroke* (line 15) to the entity cube with id *c1-thing* (line 5) from the speech utterance to achieve a finer granularity of the representation.

```
1  <utterance id="g1" confidence="76.33"
       startTime="2000" endTime="4000">
2   <gesture>
3    ...
4    <nucleus>
5     <stroke id="g1-stroke" posture="pointing"
          confidence="80.00" startTime="2200"
          endTime="2500" />
6     <hold startTime="2500" endTime="3000" />
7      ...
8    </nucleus>
9    ...
10  </gesture>
11 </utterance>
```

Figure 8: Identifiers in the representation help to refer to gestures and single gesture strokes. Probabilities can also be represented, allowing the use of Bayes and statistical classifiers which are becoming increasingly popular.

```
1  <utterance id="s1" linkedTo="g1" startTime="
       1000" endTime="5000" confidence="100.00">
2   <speech>
3    ...
4    <rel name="Patient">
5     <node id="c1-thing" pred="cube-np"
          det="dem-prox" num="sg"/>
6    </rel>
7    ...
8   </speech>
9  </utterance>
10
11 <utterance id="g1" linkedTo="s1"
       confidence="76.33" startTime="2000"
       endTime="4000">
12  <gesture>
13   ...
14   <nucleus>
15    <stroke id="g1-stroke" linkedTo="c1-thing"
          posture="pointing" confidence="80.00"
          startTime="2200" endTime="2500" />
16     ...
17   </nucleus>
18   ...
19  </gesture>
20 </utterance>
```

Figure 9: An example of a speech and a gesture utterance that are related to each other. The example shows the relation between the speech with id *s1* and the gesture with id *g1* as well as the link between the gesture stroke with id *g1-stroke* and the entity cube with id *c1-thing*.

5. APPLICATION OF MULTIML IN HUMAN-ROBOT INTERACTION

MultiML can easily be adapted to various contexts. As a proof of concept, we will show in this section, how MultiML has to be changed to meet the envisaged JAST setting. As we explained in Section 3, JAST investigates the aspects of joint-action between human and robot. To show the results of the research in JAST we are working on a scenario, in which human and robot assemble a railway sign together, which was presented in Figure 2. As mentioned

before, we wrote an XML schema file that defines the tags that are allowed in MultiML and also where and how often these tags can occur in the representation. For example, the schema specifies that each gesture unit can have several gesture phrases but only one recovery phase. We implemented this schema definition in a way so that context-independent parts of a modality can be described separately from any parts that are context-dependent.

We will describe this separation on the example of gesture representation in the JAST context. The context-dependent parts in JAST, as in other contexts, are related to gesture posture types, location descriptions, and body parts. In JAST we use a gesture recogniser that is able to classify the three gesture types *Pointing*, *Grasping*, and *HoldingOut*, which stand for gestures where the human points to objects, tries to reach an object, and holds the hand flat to show that the robot should hand over an object respectively. Figure 10 shows parts of the XML schema definition that describe these posture types. One can see from this figure how the *stroke* element (line 2) of a gesture phrase is defined with an attribute *posture* (line 5). This attribute has the type *postureType*, which is defined in the bottom part of the XML schema (lines 10–16) and can hold one of the three values for gesture postures mentioned above. This way, posture types can be generated automatically from simple configuration files, without the need of changing the whole representation language. It is also very easy to import postures and hand shapes from other gesture representation languages, such as the widely used HamNoSys.

```
1  ...
2  <xsd:element name="stroke">
3   <xsd:complexType>
4    ...
5    <xsd:attribute name="posture"
         type="postureType" />
6    ...
7   </xsd:complexType>
8  </xsd:element>
9  ...
10 <xsd:simpleType name="postureType">
11  <xsd:restriction base="xsd:string">
12   <xsd:enumeration value="Pointing" />
13   <xsd:enumeration value="HoldingOut" />
14   <xsd:enumeration value="Grasping" />
15  </xsd:restriction>
16 </xsd:simpleType>
```

Figure 10: MultiML XML schema definition for gesture posture types. Context-dependent parts are separated from context-independent parts.

The separation of context-dependent and context-independent parts of MultiML also allows the translation of other representation formats into MultiML. For example, Figure 11 shows the XML schema definition of location descriptions in the JAST domain. In JAST, locations can be described by two alternative ways, either by vague location descriptions or by more precise three dimensional coordinate vectors. The location descriptions are restricted to locations on the table in front of the JAST robot, which is due to the camera used for gesture recognition, which is mounted above the assembly table facing downwards. One can see from Figure 11 how the location, described by type *locationType* (line 1) has to be chosen from the two subtypes called *locationDescriptionType* (lines 3 and 8–14) and *locationCoordinatesType* (lines 4 and 16–22). Similar to the vague description from the JAST context in type *locationDescriptionType*, other location descriptions could be imported to MultiML, for example the descriptions introduced by Gibbon et al. [4], which are given relatively to the body of the person that produces the gesture.

```
1  <xsd:complexType name="locationType">
2   <xsd:choice>
3    <xsd:element name="locationDescription" type=
         "locationDescriptionType" />
4    <xsd:element name="locationCoordinates" type=
         "locationCoordinatesType" />
5   </xsd:choice>
6  </xsd:complexType>
7
8  <xsd:simpleType name="locationDescriptionType">
9   <xsd:restriction base="xsd:string">
10   <xsd:enumeration value="tableLeft" />
11   <xsd:enumeration value="tableMiddle" />
12   <xsd:enumeration value="tableRight" />
13  </xsd:restriction>
14 </xsd:simpleType>
15
16 <xsd:complexType name="locationCoordinatesType"
       >
17  <xsd:sequence>
18   <xsd:element name="x" type="xsd:float" />
19   <xsd:element name="y" type="xsd:float" />
20   <xsd:element name="z" type="xsd:float" />
21  </xsd:sequence>
22 </xsd:complexType>
```

Figure 11: MultiML XML schema definition for location descriptions. Locations can be either given by vague specifications like "tableLeft" or more precise by writing (xyz) coordinates.

The third adjustment that has to be made to adapt MultiML to the JAST domain is to describe the body parts that can be used to execute a gesture. In JAST this can be either the left or right hand. Figure 12 displays these body part descriptions. Similar to gesture postures and location descriptions, it is possible to import body part descriptions from other data formats easily.

```
1  <xsd:simpleType name="bodyPartType">
2   <xsd:restriction base="xsd:string">
3    <xsd:enumeration value="handLeft" />
4    <xsd:enumeration value="handRight" />
5   </xsd:restriction>
6  </xsd:simpleType>
```

Figure 12: MultiML XML schema definition for body parts that can be used to execute a gesture. In case of the JAST scenario this can be either the left or right hand.

6. CONCLUSION AND FUTURE WORK

We have introduced MultiML, a new language for representation of multimodal human utterances. MultiML can, at the time of writing this publication, represent speech and gestures. Speech representation is based on hybrid logic,

a logic formalised by OpenCCG, which is an implementation of the Combinatory Categorial Grammar formalism by Mark Steedman. Gesture representation is based on Adam Kendon's gesture units, which consist of a preparation phase, the nucleus, a phase where the actual gesture takes place, and a recovery phase. On top of these basic representations, we have added ways to represent timing information, context-dependent content, confidence values and identifiers for utterances. Additionally, we have shown how speech and gesture can be linked to form a multimodal utterance. One of MultiML's most outstanding features is its property to be easily adaptable to various contexts, which is reached by a separation of context-dependent and context-independent parts of the language. We proved this by showing how MultiML can be extended to adapt the language to a human-robot interaction in the context of the European JAST project.

In the future, we plan to parse the output of an automatic gesture classifier and generate MultiML gesture utterances from it. This representation should then be combined with speech recogniser input in a multimodal fusion component. In addition, we plan to extend MultiML so that the format can also represent other modalities, including head orientation, gaze, facial expressions, and body posture. For this extension, we want to evaluate existing formats to see if they are suitable for integration in MultiML. Candidate representation formats are for example FACS (Facial Action Coding System), which is used to describe facial expressions, or the Labanotation, a format developed by Rudolph von Laban used to represent human body movements.

Acknowledgement

This work was supported by the EU FP6 IST Cognitive Systems Integrated Project JAST (FP6-003747-IP), http://www.euprojects-jast.net/.

7. REFERENCES

[1] A. Kranstedt, S. Kopp, and I. Wachsmuth, "Murml: A multimodal utterance representation markup language for conversational agents," in *Proc. of the AAMAS Workshop on "Embodied conversational agents–Let's specify and evaluate them"*, 2002.

[2] S. Prillwitz, R. Leven, H. Zienert, T. Hanke, and J. Henning, *HamNoSys. Version 2.0; Hamburger Notationssystem für Gebärdensprache. Eine Einführung.* Hamburg: Signum, 1989.

[3] F. Landragin, A. Denis, A. Ricci, and L. Romary, "Multimodal meaning representation for generic dialogue systems architectures," in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, 2004, pp. 521–524.

[4] D. Gibbon, U. Gut, B. Hell, K. Looks, and A. T. T. Trippel, "A computational model of arm gestures in conversation," in *EUROSPEECH-2003*, 2003, pp. 813–816.

[5] J. Y. Chai, "Semantics-based representation for multimodal interpretation in conversational systems," in *Proceedings of the 19th international conference on Computational Linguistics*, vol. 1. Association for Computational Linguistics Morristown, NJ, USA, 2002, pp. 1–7.

[6] M. E. Foster, T. By, M. Rickert, and A. Knoll, "Human-robot dialogue for joint construction tasks," in *Proceedings, Eighth International Conference on Multimodal Interfaces (ICMI 2006)*, Banff, November 2006.

[7] T. Müller, P. Ziaie, and A. Knoll, "A wait-free realtime system for optimal distribution of vision tasks on multicore architectures," in *Proc. 5th International Conference on Informatics in Control, Automation and Robotics*, May 2008.

[8] P. Ziaie, T. Müller, M. E. Foster, and A. Knoll, "Using a naïve Bayes classifier based on k-nearest neighbors with distance weighting for static hand-gesture recognition in a human-robot dialog system," in *Proceedings of CSICC 2008*, Kish Island, Iran, Mar. 2008.

[9] M. Henning, "A new approach to object-oriented middleware," *IEEE Computer Society*, vol. 8, no. 1, pp. 66–75, Jan–Feb 2004.

[10] M. Giuliani and A. Knoll, "Integrating multimodal cues using grammar based models," in *Proceedings of HCI International 2007*, Beijing, China, July 2007, pp. 858–867.

[11] M. Rickert, M. E. Foster, M. Giuliani, T. By, G. Panin, and A. Knoll, "Integrating language, vision and action for human robot dialog systems," in *Proceedings of the International Conference on Human-Computer Interaction*, C. Stephanidis, Ed. Beijing: Springer, July 2007, pp. 987–995.

[12] J. F. Allen and G. Ferguson, "Actions and events in interval temporal logic," *Journal of Logic and Computation*, vol. 4, pp. 531–579, 1994.

[13] A. E. Ades and M. J. Steedman, "On the order of words," *Linguistics and philosophy*, vol. 4, pp. 517–558, 1982.

[14] M. Steedman, *The syntactic process.* Cambridge, MA, USA: MIT Press, 2000.

[15] K. Ajdukiewicz, "Die syntaktische konnexität," *Studia Philosophica*, vol. 1, pp. 1–27, 1935.

[16] Y. Bar-Hillel, "A quasi-arithmetic notation for syntactic description," *Language*, vol. 29, pp. 47–58, 1953.

[17] M. White, "Efficient realization of coordinate structures in combinatory categorial grammar," *Research on Language & Computation*, vol. 4, no. 1, pp. 39–75, 2006.

[18] J. Baldridge and G.-J. Kruijff, "Coupling ccg and hybrid logic dependency semantics," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 02)*, Philadelphia, PA: University of Pennsylania, 2002.

[19] D. McNeill, *Hand and mind: What gestures reveal about thought.* Chicago: University of Chicago Press, 1992.

[20] A. Kendon, *Gesture: Visible Action as Utterance.* Cambridge University Press, 2004.