

---

# Prototyping Multimodal Interfaces with the SMUIML Modeling Language

**Bruno Dumas**

University of Fribourg  
Boulevard de Pérolles 90  
1700 Fribourg, Switzerland  
bruno.dumas@unifr.ch

**Denis Lalanne**

University of Fribourg  
Boulevard de Pérolles 90  
1700 Fribourg, Switzerland  
denis.lalanne@unifr.ch

**Rolf Ingold**

University of Fribourg  
Boulevard de Pérolles 90  
1700 Fribourg, Switzerland  
rolf.ingold@unifr.ch

**Abstract**

In this position paper, we present an approach geared toward rapid prototyping of multimodal interfaces with SMUIML (Synchronized Multimodal User Interfaces Markup Language) and HephaisTK toolkit. The goal of SMUIML and HephaisTK is to offer developers with: on one side, a language allowing clear description of human-machine multimodal dialog and control over the way multiple input modalities have to be fused; and on the other side, an extensible tool implementing and managing this dialog.

**Keywords**

Multimodal interfaces, multimodal dialog, user-machine dialog description.

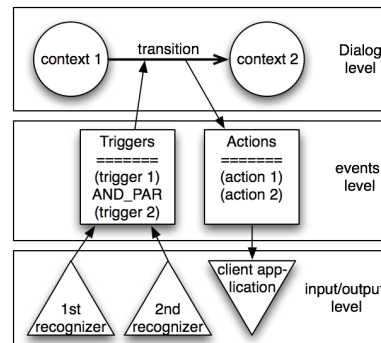
**ACM Classification Keywords**

D.2.2. Design Tools and Techniques: User Interfaces.

**SMUIML: human/machine dialog and modalities synchronization**

SMUIML has been designed from the ground up as a XML language able to describe rich interactions between human and computer. In particular, it describes every interaction between the user and the machine at three different levels, as shown in Figure 1: the lower level enables the developer using SMUIML to

describe the different modalities and associated recognizers, as well as description of incoming variables. The middle level of SMUIML describes incoming event triggers and outgoing actions, per modality. Finally the higher level of SMUIML describes the actual dialog between the user and the machine by means of a finite state machine. Synchronization of content is achieved in allowing developers to express input modality relationships with the different CARE properties (complementarity, assignment, redundancy, equivalence) presented in [5].



**figure 1:** The three levels of SMUIML.

A small instance of a SMUIML script is shown below; this instance is extracted from a use case describing a drawing table, with tangible and speech input.

```
<?xml version="1.0" encoding="UTF-8"?>
<muiml>
  <integration_description client="xpaint_client">
    <recognizers>
      <recognizer name="reactivision">
        <variable name="posx" value="x" type="int"/>
        <variable name="posy" value="y" type="int"/>
      </recognizer>
    </recognizers>
```

```
<triggers>
  <trigger name="operation">
    <source modality="speech" value="erase shape
      | rotate shape | move shape"/>
  </trigger>
</triggers>

<actions>
  <action name="draw_action">
    <target name="xpaint_client" message="draw
      $oper $shape $posx $posy"/>
  </action>
</actions>

<dialog>
  <context name="modification">
    <transition name="modif_clause">
      <par_and>
        <trigger name="operation"/>
        <trigger name="selected shape"/>
        <trigger name="position"/>
      </par_and>
      <result action=" draw_bidule"/>
    </transition>
    <transition>
      <trigger name="return"/>
      <result context="start"/>
    </transition>
  </context>
</dialog>

</integration_description>
</muiml>
```

SMUIML seeks to build on the knowledge of previous attempts at creating multimodal interaction description languages, while staying simple and expressive. Most of the approaches presented below revolve around the concept of a "multimodal web", enforced by the World Wide Web Consortium (W3C) Multimodal Interaction Activity and the proposed multimodal architecture. The work of the W3C inspired Katsurada et al. for their work on the XISL XML language [6]. XISL focuses on synchronization of multimodal input and output, as well as dialog flow and transition. As such, MUIML and XISL follow a common goal, but SMUIML tends toward a stronger versatility and readability. Another approach is the one of Araki et al. [1], who propose MIML

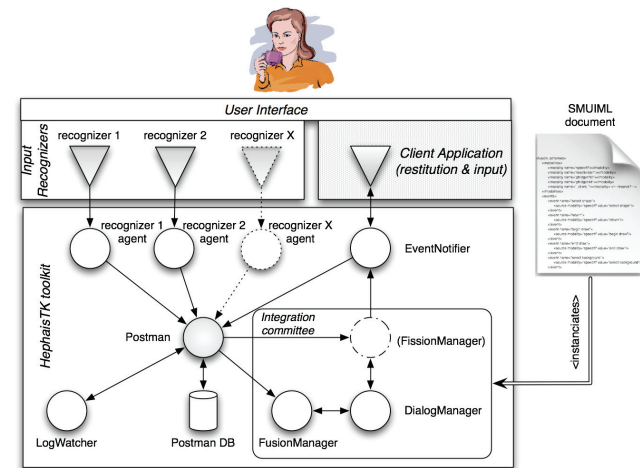
(Multimodal Interaction Markup Language). One of the key characteristics of this language is its three-layered description of interaction, focusing on interaction, tasks and platform. A similar three-layered approach has been followed for MUIML, but with a stronger accent on modality relationships and synchronization. Finally, Stanciulescu et al. [8] followed a transformational approach for developing multimodal web user interfaces based on UsiXML, also in the steps of the W3C. Four steps are achieved to go from a generic model to the final user interface. Thus, one of the main features of their work is a strong independence to the actual input and output available channels. But this versatility is at the cost of heavy preprocessing.

### HephaistTK, a toolkit using SMUIML

HephaistTK, a toolkit using SMUIML modelling language, is intended to be a toolkit allowing rapid creation of multimodal interfaces, offering a predefined set of recognizers as well as the possibility to plug into the toolkit any other modality recognizer, as long as it complies with a given set of conditions, e.g. communication with the toolkit by means of the W3C EMMA language. In the future, HephaistTK will also offer different fusion mechanisms to allow meaning from incoming recognizers to be extracted, and passed to potential client applications.

In its current state, HephaistTK is built upon a software agent system. Each time a new recognizer or synthesizer is plugged into the toolkit, an agent is dispatched to monitor it. HephaistTK uses a central blackboard architecture (see Fig. 2). A “postman” centralizes each message coming from the different input recognizers and stores it into a database. Agents interested in a specific type of message can subscribe

to the postman, which will accordingly redistribute received messages. Fusion of input modalities is achieved through meaning frames. When a Java developer wants to use HephaistTK toolkit to monitor multimodal inputs in its application, he has to declare the toolkit by means of event listeners. The fusion and dialog managers of HephaistTK are scripted by means of a SMUIML file.



**figure 2:** HephaistTK toolkit architecture.

Other researchers investigated ways to create multimodal toolkits. Bourguet [4] endeavoured in the creation of a multimodal toolkit in which multimodal scenarios could be modelled using finite state machines. A similar approach was taken with the modelization via SMUIML, but in a more modular way. Bouchet et al. [3] proposed a component-based approach called ICARE thoroughly based on the CARE[5] design space. These components cover

elementary tasks, modality-dependent tasks or generic tasks like fusion. This components-based approach has been used to create a comprehensive open-source toolkit called OpenInterface [2], in which components are configured via CIDL XML files, and a graphical editor.

### Conclusion

Of the toolkits presented above, only OpenInterface is widely available as open source software. OpenInterface has been designed as an integrated environment, in which every part of a multimodal application has to be designed, from the input to the output. The approach taken by HephaisTK and SMUIML is radically different, in that the toolkit acts more as a "plugin" to a given multimodal application, and only manages the multimodal input part and their fusion. The idea behind this is to let developers create their application with their usual tools and easing the development of the multimodal input part with easy-to-program mechanisms, as opposed to an integrated approach in which a developer would have to learn a completely new environment from the ground up. Such a modular approach allows easy prototyping of multimodal applications; it is however to be noted that creation of a full-fledged application could require a more advanced approach such as the one of OpenInterface.

To conclude, we believe in an approach with two tightly linked components: on one side human-machine dialog description by means of a XML file, and on the other side a tool implementing this dialog by means of multimodal fusion mechanisms. Finally, HephaisTK and SMUIML are still a work in progress, and will benefit in the near future from extensive validation. This

validation will take place in the context of the MeModules (<http://www.memodules.ch>) project and the Swiss NCCR IM2 project (<http://www.im2.ch>) [7].

### References

- [1] Araki, M., Tachibana, K. Multimodal Dialog Description Language for Rapid System Development. *Proc. Of the 7<sup>th</sup> SIGdial Workshop on Discourse and Dialogue*, Sydney, Australia, July 2006.
- [2] Benoit, A., Bonnaud, L., Caplier, L., Damousis, I., Tzovaras, D., Jourde, F., Nigay, L., Serrano, M., Lawson, J.-Y. Multimodal Signal Processing and Interaction for a Driving Simulator: Component-based Architecture. In *JMUI*, Vol 1, No 1 (2007).
- [3] Bouchet, J., Nigay, L., and Ganille, T. ICARE Software Components for Rapidly Developing Multimodal Interfaces. In *Proc. of ICMI'2004*, State College, Pennsylvania, USA, October 2004.
- [4] Bourguet, M. L. A Toolkit for Creating and Testing Multimodal Interface Designs. In *proc. of UIST'02*, Paris, Oct. 2002.
- [5] Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J. and Young, R. Four Easy Pieces for Assessing the Usability of Multimodal Interaction: The CARE properties. In *Proc. of INTERACT'95*, Lillehammer, Norway, June 1995.
- [6] Katsurada, K., Nakamura, Y., Yamada, H., and Nitta, T. 2003. XISL: a language for describing multimodal interaction scenarios. In *Proc. of ICMI 2003*, Vancouver, British Columbia, Canada, Nov. 2003.
- [7] Lalanne, D., Evequoz, F., Rigamonti, M., Dumas, B., Ingold, R. AAn ego-centric and tangible approach to meeting indexing and browsing. In *Proc. of MLMI'07*, Brno, Czech Republic, July 2007.
- [8] Stanciulescu, A., Limbourg, Q., Vanderdonck, J., Michotte, B., and Montero, F. 2005. A transformational approach for multimodal web user interfaces based on UsiXML. In *Proc. of ICMI'05*, Torento, Italy, Oct. 2005.