Extending multimedia languages to support *multimodal-multiuser* interactions

Álan L. V. Guedes
TeleMídia/PUC-Rio
alan@telmidia.puc-rio.br

Simone D. J. Barbosa
Informatics Departament
simone@inf.puc-rio.br

ABSTRACT

Hardware and software technologies have given rise to a new class of human-computer interfaces that both explores multiple modalities and allows for multiple collaborating users. When compared to the development of traditional *single-user WIMP* (windows, icons, menus, pointer)-based applications, however, applications supporting the seamless integration of *multimodal-multiuser* interactions bring new specification and runtime requirements. In this thesis, with the aim of assisting the specification of multimedia applications that integrate multimodal-multiuser interactions, we: (1) propose the MMAM (Multimodal-Multiuser Authoring Model); (2) present three different instantiations of it (in NCL, HTML, and a block-based syntax); and (3) evaluate the proposed model through a user study. MMAM enables programmers to design and ponder different solutions for applications with multimodal-multiuser requirements. Its instantiations served as proofs of concept about the feasibility of our model and enabled us to perform the user study. The user study focused on capturing evidence of both the *user understanding* and the *user acceptance* of the proposed model. 94.47% of the participants gave positive answers to the block-based representation TAM questions, whereas 75.17% of them gave positive answers to the instantiations-related questions.

KEYWORDS

Multimedia Languages, Multimodal User Interactions, MUI, Multiuser User Interactions, NCL, HTML

1 INTRODUCTION

This article presents a summary of the results obtained in the Thesis[1] defended in the PUC-Rio Postgraduate Program in Informatics at September 29, 2017. The work was developed by Alan L.V. Guedes, under the guidance Simone D. J. Barbosa and Luiz F. G. Soares (*in memoriam*).

We naturally interact with our physical surroundings and other human beings through multiple senses (e.g., sight, hearing, and touch) and using different communication modalities (e.g., voice, writing, and gestures) [8]. Also, we usually experience those multiple communication modes simultaneously and make sense of our environment through their combination. For instance, during a conversation, audio cues allow us to identify a speaker's identity and location, while the conversation itself is commonly extended with gestures. Aiming to support more natural human-computer interfaces —and driven by recent advances in hardware and software technologies —, traditional *single-user WIMP* (windows, icons, menus, pointer)-based multimedia applications are evolving towards more advanced interfaces that support both *multimodal* and *multiuser* features.

A *Multimodal User Interface* (MUI) processes two or more combined user input modalities (e.g., speech, pen, touch, gesture, and head and body movements) in a coordinated way with output modalities [8]. An *input modality* is a mode of communication that conveys information generated by human communication activities (e.g., speech and gestures) and captured by input devices or sensors (e.g., microphone, pen, and motion sensors). An *output modality* is a mode of communication that conveys stimuli to be perceived by the human senses (e.g., hearing and vision). Essential features of MUIs are illustrated by Bolt's "Put-That-There" seminal work [1]. In the proposed system, the user interacts with the system through gestures and voice commands to plan tactical activities of military units on a map. The user can move a military unit by first pointing his/her finger to a unit on the battlefield while saying "put that"; and then pointing his/her finger to the desired location and saying "there."

Multiuser Interfaces (sometimes called *Identity-aware User Interfaces* (IAUI) are those in which the system can distinguish (and usually the programmer is aware of) the different users or groups of users who are interacting with the system. [7]. Computer games and computer-supported cooperative work (CSCW) have been fertile environments for multiuser interfaces. In computer games, different users are usually identified by different joysticks (different devices), whereas in CSCW systems, many different mechanisms allow the identification of the users interacting with the system, who can usually work together towards a specific goal. Only increasing the number of users, however, does not necessarily imply that the system is aware of and enables programmers to distinguish the different inputs/outputs from/to different users. For instance, shared screen applications can be viewed as a primitive multiuser systems, since it is possible to have multiple interacting users. In general, however, those systems are not aware of the different interacting users (or group of users) and handle all the inputs as coming from a single entity.

When compared to traditional multimedia/single-user applications, *multimodal-multiuser* bring new requirements from both *system* and *specification* perspectives. On the one hand, from a system perspective, there are new requirements, such as the support for a fine-grained synchronization between audiovisual media objects and the *multimodal-multiuser* interactions, possibly coming from different (local or remote) devices. On the other hand, from a specification perspective, a flexible system should support abstractions so that programmers can easily create new applications combining

[1] https://doi.org/10.17771/PUCRio.acad.32313

In: I Concurso de Teses e Dissertações (CTD 2019), Rio de Janeiro, Brasil. Anais Estendidos do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia). Porto Alegre: Sociedade Brasileira de Computação, 2019.
ISSN 2596-1683

different input/output modalities (possibly) coming from/going to different users (or group of users).

In our current research effort, we focus mainly on the *specification* issues of *multimodal-multiuser* systems. More specifically, we aim to explore how such support can be achieved by extending current declarative multimedia languages such as HTML5 (HyperText Markup Language)[2] and NCL (Nested Context Language)[3]. It is worth mentioning that, although such languages (usually based on XML) focus on providing high-level abstractions for the definition of media items as part of a multimedia application and their presentation in time and space, most of them *lack support to multimodal features*, and none of them support abstractions to *specify multiple interacting users or group of users*.

Some related work have discussed the addition of multimodal-only features in multimedia languages. However, in this thesis, we propose a reusable model called *Multimodal Multiuser Authoring Model (MMAM)*, which provides a set of first-class concepts to be supported by declarative multimedia languages and allows a seamless integration of *multimodal-multiuser* features. We have also provided two new instantiations of the proposed model: (1) an HTML5-based one, allowing more developers to use the proposed concepts; and (2) a *block-based* one, which helped us to evaluate the proposed abstractions. Moreover, we evaluate MMAM through a user study aimed at capturing evidences of both the *user understanding* and the *user acceptance* of the model. The study was performed by asking developers to perform some tasks and then applying a Technology Acceptance Model (TAM) [2] questionnaire when using block-based syntax and the extended versions of NCL or HTML. The results of the user study indicate that developers understood the model and found it both useful and easy to use.

The remaining of this paper the structured as follows. Section 2 presents MMAM, highlighting its main features. Section 3 discusses how MMAM concepts can be incorporated into concrete syntaxes. The user study and its results are discussed in Section 4. Finally, Section 5 presents our final remarks.

2 MMAM'S ENTITIES

Figure 1 illustrates the main MMAM entities that aim to satisfy the requirements above. Some of them are already partially discussed in some research fields. For instance, *Media* and *Recognizer* are discussed in multimedia documents and multimodal interaction fields, respectively. But our work is the first to unify them in the *multimodal-multiuser* context.

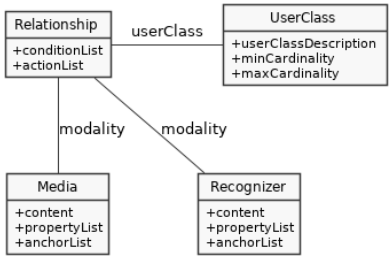

Figure 1: Simplified MMAM class diagram

[2]https://www.w3.org/html
[3]http://handle.itu.int/11.1002/1000/12237

Media aims at modeling **(1) unimodal output modalities**. This entity is an abstraction for producing output in certain modalities through audiovisual devices and actuators. A media entity is defined by a *content*, a list of *anchor*s, and a list of *properties*. The media *content* enables the output modality semantics to be decoupled from the application structure. It can be an ordinary audiovisual content —e.g., audio, video, and image— or a document described in a unimodal synthesizer language, e.g., an SSML[4] for voice synthesis or SEDL[5] for sensorial effects. *Anchor*s are portions of the content presentation. For instance, an *anchor* may define a temporal portion of a video (*e.g.*, a segment of the video ranging from 30 to 50 seconds). In this case, the anchors are automatically activated/deactivated as the presentation content of the *Media* reaches the content portion related to each *Anchor*. Additionally, an *anchor* may refer to the identifier of an internal element in the synthesizer document, *e.g.*, an SSML element to be synthesized. The activation of such anchor means that the *Media* is rendering the content related to that *anchor*. *Properties* define parameters for *Media*, such as the transparency of video, the volume of the audio output produced by an SSML speech synthesizer, and the frame rate of an avatar rendering.

Recognizer is an abstraction for identifying an expected input in a certain modality captured from an input device or sensor. It aims at modeling the **(2) unimodal input modalities**. A recognizer entity is defined by a *content*, a list of *anchors*, and a list of *properties*. The recognizer *content* enables the input modality semantics to be decoupled from the application structure. Its *content* can be a unimodal recognizer description, *e.g.*, an SRGS[6] file for voice or GML[7] for gesture recognition. Its *anchor*s are portions of the content. They define when to expect, within a document, a certain input modality to be recognized. For instance, an SRGS defines one rule element for each expected speech input (*e.g.*, a certain rule may define that acceptable input tokens are "put that" and "there"). An *anchor* enables the multimedia document to know when the recognition of a token is complete. *Propertie*s define parameters for the dynamic values inside the *Recognizer*. For example, an ink recognizer can have properties to indicate the current position of the ink pointer.

Relationship aims at specifying the **(3) behavior and synchronization between input and output modalities**. It synchronizes the modalities perceived by users through *Media* and the modalities generated by users through *Recognizers*. A *Media* or *Recognizer* *anchor* represents synchronization points during their activated period. Then, the *Relationship* entity enables developers to specify the synchronization among those anchors. In the "Put-That-There" scenario, for instance, *Relationship* entities express the combined use of speech synthesis and visual objects (*Media*) with the speech and gesture recognition (*Recognizers*). This synchronization is defined by a set of *conditions* and *actions*. When the *conditions* are satisfied, the *actions* are performed.

Conditions can be combined in *CompoundCondition*s. For instance, a *Relationship* can be defined to be triggered when portions of a video are activated and the recognition of a sentence is complete. The supported operators are *or*, *and* and *seq*. In the "Put-That-There"

[4]http://w3.org/TR/speech-synthesis11/
[5]http://iso.org/standard/65396.html
[6]http://w3.org/TR/speech-grammar/
[7]http://gestureml.org

scenario, for instance, authors must use a *seq* operator to guarantee that the interactions must occur in the specified order (first, the "put that"; then, the "logo" selection, etc.).

The *UserClass* entity is used to support *multiuser* features in MMAM. It models the different roles a user, **(4) group of users**, can play when interacting with an application. A *UserClass* is defined by its *minCardinality* and *maxCardinality* attributes, and by a *userClassDescription*. The cardinality attributes limit the number of users that can be part of a specific *UserClass*, and can be either an integer or the "undefined" keyword. The *userClassDescription* enables the UserClass semantics to be decoupled of the application structure. It specifies how users are identified as being part of the *UserClass* given a required **(5) user profile**. To do that, each user who can interact with the system must have a *profile description*. For instance, the user profile may include information such as whether he/she is a student or a professor; or it may include the devices he/she is using to interact with the application. The *userClassDescription* should then specify which users, based on their profiles, are part of each *UserClass*. The abstract MMAM does not prescribe the *userClassDescription* syntax details, which should be defined by a specific instantiation of the model. In the current provided instantiations, we use an RDF (Resource Description Framework)[8] description for user *profile description*, and *SPARQL*[9], which enables queries over RDF entities.

To support multiuser-oriented development, the recognition *condition* in a *Relationship* must be parameterized with a *userClass* attribute. It enables the specification of **(6) which specific users (or group of users) are involved in each input/output modality interaction**. More precisely, the value of this attribute defines which specific user from a *userClass* can be responsible for generating the event. By doing so, it is possible to limit which specific users can trigger a *Relationship*.

3 MMAM'S INSTANTIATIONS

We instantiated the *MMAM*'s entities in the NCL 3.0, HTML, and in a block-based representation. NCL is an XML-based language, then to instantiate the MMAM's entities, we extended its XML schemes. The web browser vendors follow their own HTML markup language[10]. Then, to extend HTML, we use the browser vendor standard *CustomElements*[11], which provides a JavaScript API to extend the HTML markup. Thus, the extensions below can be used directly in HTML5 pages among other elements and with current web browsers. And the block representation was developed using the web-based version of the Blockly[12] framework.

NCL 3.0 and HTML5 are multimedia languages. The first is focused on interactive video content and the former focused mainly on supporting text-based interactive content, which includes support for audio and video content. They do not provide a seamless integration of new input modalities, besides the WIMP-based ones, neither are they aware of the (groups of) users interacting with the application. Table 1 presents the MMAM's entities in the extended NCL and HTML syntaxes. The bold elements indicate the new elements proposed.

Regarding the block-based representation, a *Media* or a *Recognizer* entity is defined by joining the corresponding block with a content block, which can be an image, audio, video or speech synthesis block. A *UserClass* is defined by joining one UserClass block, with the cardinality fields filled in, and device blocks, which can be microphone or gesture sensor. Then, a *Relationship* is defined by joining one *Relationship* block with conditions and action blocks, with target block fields filled in. *Simple condition block* can define the trigger for, for instance, the beginning or end of a media/recognizer anchor and the recognition of a recognizer anchor or block id. Figure 2 illustrate some of the implemented blocks representations.

MMAM's entity	NCL	HTML
Media	`<media>`	`<img>`, `<audio>`, `<video>`, and **`<mm-media>`**
Recognizer	**`<input>`**	**`<mm-input>`**
Relationship	`<link>`	**`<mm-link>`**
UserClass	**`<userClass>`**	**`<mm-userClass>`**

Table 1: *MMAM*'s entities instantiation

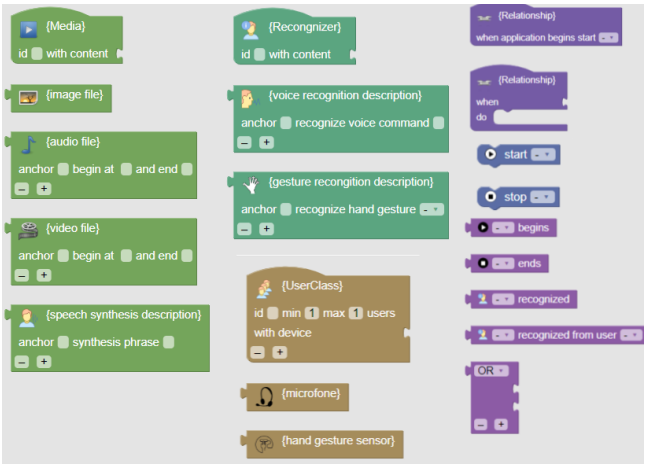

Figure 2: Some block-based representations elements

4 USER STUDY

The main advantage of the presented MMAM instantiations is that they are well integrated with current multimedia languages, and thus can be readily learned by programmers who know these languages. However, when using one of those instantiations, evaluating the proposed model alone *(i.e.*, without the interference of the syntax and semantics of HTML5 or NCL 3.0) is challenging. Thus, as a way to help us focus on the concepts proposed by MMAM during the user study (discussed in the next section) we also provide a block-based syntax for it. Although block-based representations also contain their own syntax, they help users to abstract from specificities and lower-level textual syntax of the languages. Indeed, due to this feature such a paradigm is commonly used for teaching programming (*e.g.*, Scratch[13]) or in code generation tools.

To evaluate MMAM, we conducted a user study to capture both *user understanding* and *acceptance* of the proposed concepts. To

[8] http://w3.org/TR/REC-rdf-syntax
[9] http://w3.org/TR/rdf-sparql-query
[10] https://html.spec.whatwg.org
[11] http://html.spec.whatwg.org/multipage/custom-elements.html#custom-elements
[12] http://developers.google.com/blockly
[13] http://scratch.mit.edu

capture the *user understanding* of the model, we performed a task-based study, in which the participant received examples of MMAM applications and were asked to perform some changes. To capture the *user acceptance* of the MMAM concepts, we performed a study based on the TAM [2], which allows us to capture the *perceived usefulness* (PU) and *perceived ease of use* (PEOU) of the model. We asked the participants' opinion through statements based on TAM, namely on Gefen's and Keil's work [2].

In the first part, the study focuses on the conceptual entities (using the block-based representation). Then, the participant is guided to the first part of the study, which targets all the participants. It introduces each MMAM entity, shows usage examples, and asks the participant to perform tasks using the block representation. Two tasks ask the participant to describe a given solution which used *Media*, *Recognizer* and *Relationship* entities. Two other tasks ask him/her to develop a solution using CompoundCondition and *UserClass*. We then asked the participants' opinion about the MMAM concepts.

The second part focuses on one of the concrete textual syntaxes of the model (NCL or HTML), chosen based on his/her stated knowledge of each language. If a participant knows more about HTML, he/she will be presented with questions about the MMAM HTML syntax, otherwise he/she will be presented with the MMAM NCL syntax. For each MMAM entity, it introduces concepts, shows examples, and asks the user to perform tasks using the extended language. Then we ask the participants' opinion on TAM statements about the extended language. At the end, the form asks the participant's opinion regarding the quality of our instantiation, through the statement "The concepts presented in the previous section are clearly instantiated in the extended language."

The user study involved 37 participants: 35 are graduate students with a background in Computer Science, and 2 are design graduate students. Based on the participants' self-reported knowledge of NCL and HTML, we ended up with 21 participants in the HTML group and 16 in the NCL group. They mainly consider themselves as skilled (34 moderate to expert answers) in their group language.

In general, both NCL and HTML participants performed the tasks with no errors (8 from NCL and 13 from HTML) or minor errors (8 from NCL and 12 from HTML). This indicates that they understood the model fairly well. If participants had performed the tasks with more errors, it would indicate that they had not clearly understood the proposed entities, and the TAM answers would not be sufficiently informative. However, in general, both NCL and HTML participants performed the tasks, on the block-based and extended language representations, with few errors, so their TAM answers can be considered well informed.

The participants' answers to the TAM statements about the block-based representation, and the results of the statements about the extended language syntax (NCL or HTML). On average, 94.47% of participants gave positive answers (*i.e*, Agree a little, Agree, or Strongly Agree) regarding the block-based representation, whereas 75.17% gave positive answers regarding the extended language. Based on such answers, it is possible to conclude that the study participants found the model both *useful* and *easy to use*.

Although most users said that the entities were clearly instantiated, their answers to the TAM questions showed a small difference between the block-based and extended languages representations. The participants gave slightly more positive answers for the block-based representation. To understand why this was the case we need to conduct further studies.

5 FINAL REMARKS

By providing a high-level model for authoring multimedia applications that support *multimodal-multiuser* features, the main contribution of this thesis is to support the development of more natural and cooperative scenarios —e.g., the "I-Get-That-You-Put-It-There" extension to Bolt's classic scenario— in the scope of declarative multimedia languages. In particular, based on the results of the user study discussed in this thesis, we may conclude that the proposed model is easy to *understand* and can be well *accepted* by the target users. Furthermore, the proposed instances of MMAM in a multimedia language allow the representation of applications in which the content for recognizers and synthesizers can be handled in an agnostic way and are decoupled from the behavior of the application, which is a good modeling practice in general.

Regarding the impact of this Thesis, Table 2 presents the main archived publications. Among then, we highlight a Spring Multimedia Tools and Applications Journal.

Paper	Publication	Citations[14]
[4]	WebMedia Main Track 2015	1
[9]	ACM DocEng Doctoral Consortium 2016	0
[5]	WebMedia Main Track 2016	3
[4]	WebMedia WSoT 2016	4
[3]	Springer Multimedia Tools and Applications 2017	3
[6]	ACM DocEng Main Track 2019	0

Table 2: Publication Impact.

REFERENCES

[1] Richard A. Bolt. 1980. Put-That-There: Voice and Gesture at the Graphics Interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*. ACM, New York, NY, USA, 262–270. https://doi.org/10.1145/800250.807503

[2] David Gefen and Mark Keil. 1998. The Impact of Developer Responsiveness on Perceptions of Usefulness and Ease of Use: An Extension of the Technology Acceptance Model. *SIGMIS Database* 29, 2 (April 1998), 35–49. https://doi.org/10.1145/298752.298757

[3] Álan L. V. Guedes, Roberto G. de A. Azevedo, and Simone D. J. Barbosa. 2017. Extending multimedia languages to support multimodal user interactions. 76, 4 (2017), 5691–5720. https://doi.org/10.1007/s11042-016-3846-8

[4] Álan L. V. Guedes, Marcio Cunha, Hugo Fuks, Sérgio Colcher, and Simone Diniz Junqueira Barbosa. 2016. Using NCL to Synchronize Media Objects, Sensors and Actuators. http://www.lbd.dcc.ufmg.br/colecoes/wsotwebmedia/2016/003.pdf

[5] Álan L. V. Guedes, Roberto G. de A. Azevedo, Sérgio Colcher, and Simone D.J. Barbosa. 2016. Extending NCL to Support Multiuser and Multimodal Interactions. In *Proceedings of the 22Nd Brazilian Symposium on Multimedia and the Web (Webmedia '16)*. ACM, 39–46. https://doi.org/10.1145/2976796.2976869

[6] Álan L. V. Guedes, Roberto G. de A. Azevedo, Sérgio Colcher, and Simone D.J. Barbosa. 2019. Modeling Multimodal-Multiuser Interactions in Declarative Multimedia Languages (accepted to be published). In *Proceeding of the eighth ACM symposium on Document engineering - DocEng '19*. ACM.

[7] Cornelia Haber. 2001. Modeling Multiuser Interactions. In *Proceedings at the First European Computer Supported Collaborative Learning Conference, Maastricht, Germany*. 22–24.

[8] Matthew Turk. 2014. Multimodal interaction: A review. *Pattern Recognition Letters* 36 (2014), 189–195. https://doi.org/10.1016/j.patrec.2013.07.003

[9] Álan L.V. Guedes. 2016. Towards supporting multimodal and multiuser interactions in multimedia languages. In *Proceeding of the eighth ACM symposium on Document engineering - DocEng '16: Doctoral Consortium*. https://doceng2016.cvl.tuwien.ac.at/?page_id=594

[14] https://scholar.google.com