Ontology-driven Multimodal Interface Design for an Emergency Response Application

Elena Tsiporkova
Sirris Software & ICT Group
Diamant Building, A. Reyerslaan 80,
B-1030 Brussels, Belgium
elena.tsiporkova@sirris.be

Tom Tourwé
Sirris Software & ICT Group
Diamant Building, A. Reyerslaan 80,
B-1030 Brussels, Belgium
tom.tourwe@sirris.be

Nicolás González-Deleito
Sirris Software & ICT Group
Diamant Building, A. Reyerslaan 80,
B-1030 Brussels, Belgium
nicolas.gonzalez@sirris.be

Anna Hristoskova
Ghent University, Department of Information Technology – IBBT, Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, Belgium
anna.hristoskova@intec.ugent.be

ABSTRACT

In this paper, we propose an ontology-driven modelling framework, which allows to capture the domain and expert knowledge available within the interface design community, and to support designers in their daily design tasks by eliciting user and application dependent design recommendations. We illustrate how this framework can be used in practice with a concrete case study devoted to multimodal interface design for the purpose of emergency response applications.

Keywords

ontology, data modelling, emergency response, decision support, context-awareness

INTRODUCTION

The development of multimodal applications is inherently complex due to the fact that these applications are usually targeting complex data-rich environments and need to address challenges as data overload, requirements for improved recognition performance, support for time and attention sharing, etc. [1]. Moreover, in order to use the best suitable modality at a given time, the application must also be context-aware. The challenge is to design multimodal interfaces which can reliably interpret a continuous input from different visual, auditory, and other sources in order to make an accurate context assessment and response planning in support of the user's tasks.

Several authors worked on establishing formal principles for multimodal user interface design, following principles of user-centered design philosophy. Reeves et al. [2] defined a set of principles divided in six different categories of guidelines: requirements specifications, designing multimodal input and output, adaptability, consistency, feedback and error preventions/handling. Some of the included principles are: design for the broadest range of users and contexts of use, address privacy and security, maximise human cognitive and physical abilities, integrate modalities in a manner compatible with user preference, context, and system functionality. Although these principles represent a valuable methodological advancement in the domain of user-centered multimodal interaction design, they are of a little practical use to the daily activities of the designers since a considerable gap exists between the theory (formal guidelines) and the practice of multimodal human interface design, as different experts might approach the same interface design tasks in different ways based on personal expertise, background and intuition.

Our aim in this article is to work toward bridging this gap via the application of semantic technologies (e.g. ontologies) for capturing the available domain and expert knowledge in the field of multimodal interface design. There are several advantages associated with such an approach: it guarantees a uniform approach across different designers within the same organisation, allows for semantic inter-usability of the formal guidelines across different applications and domains, facilitates context representation, and is open to allow for knowledge evolution and growth. It is illustrates how this semantic framework can be used in practice with a concrete case study devoted to multimodal interface design for the purpose of emergency response applications.

LEVELS OF MODELLING ABSTRACTION

We propose here a semantic modelling framework (see [3]), which allows to capture general *domain knowledge* and *expert knowledge*. The former considers all factual information relevant to the Human-Computer Interaction (HCI) domain, while the latter attempts to capture the HCI community's available and well established guidelines and best practices related to multimodal application design. Both domain and expert knowledge are described via an ontology, a formal representation of knowledge by a set of key domain concepts and the relationships between those concepts. We complement this with *application-specific knowledge* and illustrate how the framework supports the decision-making of which modalities are suitable candidates for an application.

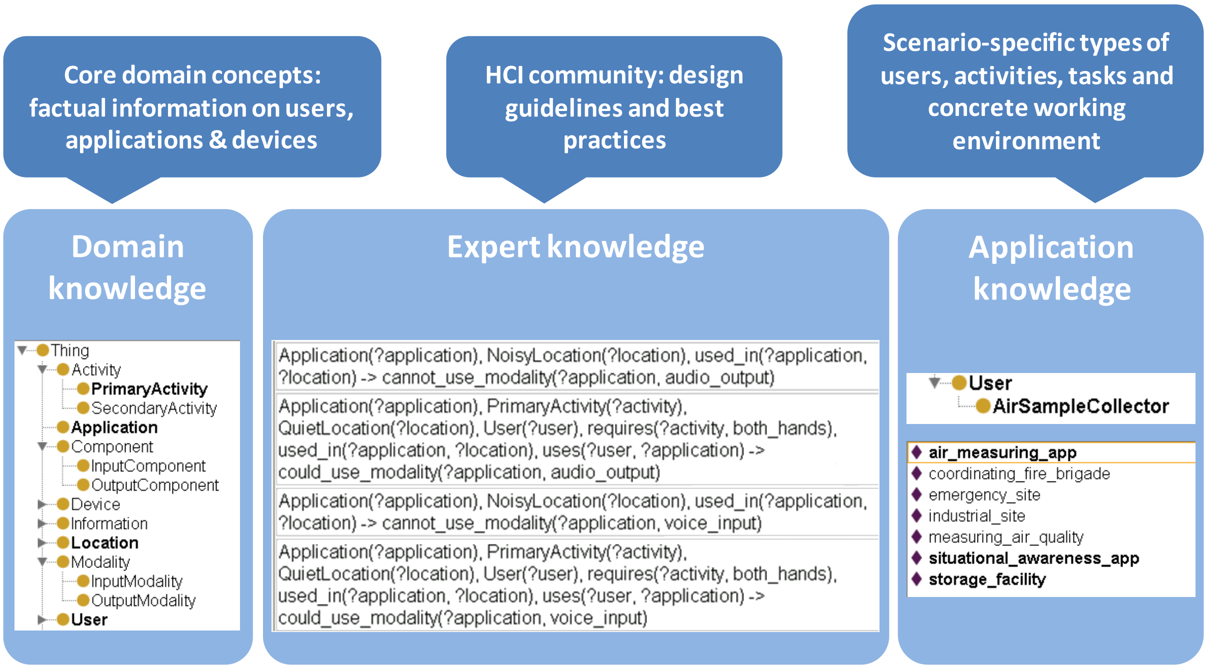

Figure 1: Schematic overview of the different levels of semantic modelling abstraction: domain, expert and application knowledge

The proposed method structures the semantic modelling in three levels of abstraction as presented in Figure 1. The first two levels, domain and expert knowledge, are modelled within our "core" HCI ontology, while the application-specific knowledge is defined in an additional application-specific ontology, which is an extension and instantiation of the core ontology and reflects the concrete context of use of the application.

HCI ONTOLOGY

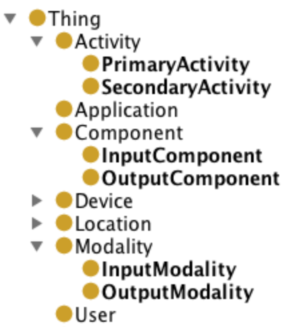

Figure 2: High-level domain concepts in the semantic modelling framework.

We define the core HCI ontology, consisting of general and high-level key domain concepts as depicted in Figure 2. For instance, the class **User** represents a user of an application, which itself is represented by the class **Application**. The class **Device** represents the device that the user is using and on which applications run, while the class **Component** represents the different components of a device. The latter class is further specified as being either an **InputComponent** (e.g. a microphone) or an **OutputComponent** (e.g. a speaker). Different components support a different **Modality** (e.g. a microphone supports voice input). The class **Activity** represents the activities that a user can engage in, subdivided into **PrimaryActivity** and **SecondaryActivity**.

These concepts are related through the relationships described in the table below:

Name	Specifies…	Properties
is_located_in	a user is located in a location	functional, inverse property: **contains**
uses	a user uses an application	inverse property: **used_by**

performs	a user performs an activity	
runs_on	an application runs on a device	
has_component	a device features a component	
has_noise_level	the noise level of a location	functional
has_access_to	an application has access to a component	defined as property chain: **runs_on** and **has_component**
used_in	an application is used in a location	defined as property chain: **used_by** and **is_located_in**
requires	an activity requires a capability	
supports_activity	an application supports performing an activity	defined as property chain: **used_by** and **performs**
supports_modality	a component supports a modality	
has_property	a user has a characteristic	

We model *domain knowledge* by specifying necessary conditions for the key domain concepts in our core ontology. We consider domain knowledge to be any factual information about users, applications and devices that potentially influences the decision about which modality to provide. This includes obvious information such as the specific input/output modality supported by a component of a device, but also information such as physical and social aspects of the user's working environment, or particular aspects of the nature of the activity (e.g. primary and secondary tasks).

For example, a contemporary computer features a microphone and speakers, which we model by defining two necessary conditions on a class **PersonalComputer**, a subclass of class **Device**, as follows:

has component value *microphone*
has component value *speakers*

where microphone is an instance of class **InputComponent** (supporting the voice input modality), and where speakers is an instance of **OutputComponent** (supporting the audio output modality). Modelling the class **PersonalComputer** in this way, we formally define that any personal computer in our domain necessarily includes both a microphone and speakers, and hence necessarily supports voice input and audio output modalities.

As an example of information regarding the user's environment, we define a subclass **NoisyLocation** of class **Location** with a necessary condition stating **has_noise_level value** *loud.*

Expert knowledge is understood as a set of design guidelines which capture the expertise and experience of the HCI practitioners. They describe applicability conditions and constraints for the use of a particular multimodal interface. We capture design guidelines via the Semantic Web Rule Language (SWRL) [4], which is used for coding procedural knowledge in ontologies in the form of rules. This allows existing description logic reasoners such as Pellet [5] to execute data transformations defined in SWRL rules.

The main idea is to have the reasoner derive extra properties stating whether or not an application can use a particular modality: **could_use_modality** and **cannot_use_modality**. For instance, having in mind that the accuracy of voice technology is heavily dependent on environmental noise conditions (e.g. background noise), we can rule out interaction with an application through vocal commands and audio output if the user is using the application in a noisy environment. This can be expressed as follows in SWRL:

Application(*?application*), **NoisyLocation**(*?location*), **used_in**(*?application, ?location*) →
cannot_use_modality(*?application, audio output*),
cannot_use_modality(*?application, voice input*)

MULTIMODAL INTERFACE DESIGN FOR EMERGENCY DISPATCHING APPLICATIONS

In the context of a large EU project called ASTUTE[i], an emergency management demonstrator is being developed considering a decentralized solution where the emergency workers are equipped with portable or embedded devices capable of receiving, sending, and visualizing dispatching events and context information such as annotated geographical maps. The emergency workers collaborate within their task force and between different units backed by a central dispatching room. A map-centric user interface provides the field workers with a clear and up-to-date overview of all events including all other operating and idle units. It is expected that offloading and distributing the dispatching tasks should greatly enhance the situational awareness during stressful events.

Thus an environment is targeted where the user is surrounded by a multitude of devices through which he can interact with different applications that support him in his activities and tasks. The main goal of the HCI (human-computer interface) design task is to enable applications to adapt to changing situational contexts, i.e. to send the right information at the right moment in time, through a device that offers the optimal output modality for that information as well as an appropriate input modality to allow the user to react.

We consider two concrete application scenarios derived from the emergency management demonstrator of the ASTUTE project. This demonstrator considers a fire in an industrial site, and involves the coordination of all the relevant stakeholders in order to evacuate the site, extinguish the fire and bring the area affected by the fire back to a usable state.

In this context, our original core ontology needs to be complemented and extended by creating an ontology with relevant application-specific knowledge. Examples of such application-specific knowledge are the different types of users involved in this scenario (fire fighters, fire commanders, fire station dispatchers, air sampling collectors, emergency communication managers, medical experts, company employees, etc.), their activities and tasks (fire fighting, locating water supplies, rescuing company employees that could not leave a building, logging relevant information, defining security perimeters in the presence of dangerous substances, etc.), and the concrete working environment they are located in (an administrative office where the fire started, a storage facility with smoke and high temperatures, outside a building where dangerous substances might be being spread in the air, inside a medicalised tent, etc.).

Our two concrete scenarios involve two rather different types of stakeholders, in terms of role, context and needs:

- *An air sampling collection team* that needs adequate support to perform optimally its activities in the field around the fire location;
- *A fire brigade officer* who is coordinating the firemen fighting the fire emergency and communicating with the dispatching control room.

The air sampling collection team frequently measures the quality of the air, its speed and direction, as well as other weather conditions at different locations around the industrial site in order to evaluate how dangerous substances are actually being spread. Members of this team keep a record of the measurements in an application running on a mobile device. Due to regulations, they are required to wear gloves and a mask while performing the measurements. Finally, measurements take place at locations sufficiently far away from the location of the fire so that the working environment of the members of this team can be considered most of the time as quiet. We model this application-specific knowledge by:

- defining an instance of **PrimaryActivity** called *measuring_air_quality*, which **requires** the use of *both_hands*;
- defining an instance of the class **Tablet** (a subclass of **Device** which features a *microphone*) called *nicolas_tablet*;
- defining an instance of **Application** called *air_measuring_app*, which **runs_on** *nicolas_tablet*;
- defining an instance of **OutdoorLocation** called *industrial_site*, which **has_noise_level value** *quiet*;
- defining a subclass of **User** called **AirSampleCollector** with necessary conditions stating that each instance **performs** the *measuring_air_quality* activity, **uses** the *air_measuring_app*, and **is_located_in** an *industrial_site*;
- defining an instance of **AirSampleCollector** called *nicolas*.

The semantic engine can now combine this knowledge with the domain-specific knowledge and the expert knowledge to automatically suggest that voice could be used as input modality. It does so by deriving that the relationship **could_use_modality** holds between the *air_measuring_app* application and the *voice_input* modality.

The air sampling collector also uses the air sampling record application while back in his office to perform some statistical analysis on the data and produce a formal report. In such circumstances, the air sampling collector will certainly choose to use traditional interface modalities like keyboard and mouse. This can be accordingly coded in the ontology.

In our second scenario, we consider a fire brigade officer situated at the emergency site. The fire brigade officer is coordinating the firemen fighting the fire and communicating with the dispatching control room. She is moving around the site, carrying a mobile device that is running an application supporting situational awareness, allowing her to be aware of what is happening and helping her decide what is the appropriate course of action. Understandably, the emergency site is quite noisy, as people deploy heavy materials, shout instructions to each other, find themselves in a stressful situation, etc. It is thus logical that vocal and audio technologies are excluded as potential interface modalities for the situational awareness application. We model this application-specific knowledge as by:

- defining an instance of **PrimaryActivity** called *coordinating_fire_brigade*, which **requires** *no_hands*;
- defining an instance of the class **Application** called *situational_awareness_app* which **supports_activity** *coordinating_fire_brigade*;
- defining an instance of **Location** called *emergency_site*, which **has_noise_level** value **loud**;
- defining **FireBrigadeOfficer** as a **User** who **has_property** *mobile*, who **performs** the *coordinating_fire_brigade* activity, who **uses** the *situational_awareness_app* application, and who **is_located_in** the *emergency_site*;
- defining an instance of **FireBrigadeOfficer** called *elena*;

With this additional application knowledge, the semantic engine can automatically derive that haptic input (i.e. touch) could be considered as a modality, by deriving that the **could_use_modality** holds between the *situational_awareness_app* application and the *haptic_input* modality. In addition, the engine derives that voice input and audio output modalities cannot be used, due to the fact that the officer is working in a **NoisyLocation**, and that the manual input (e.g. by means of a keyboard) cannot be used, because the officer needs to be *mobile*.

CONCLUSION

This paper presents an initial attempt to formally model and exploit relevant HCI domain knowledge and practitioners' expertise in support of selecting appropriate modalities during the human-machine interface design process. The framework will be further validated in concrete emergency dispatching application scenarios.

ACKNOWLEDGMENTS

This research is supported by the ARTEMIS JU and IWT (www.iwt.be) through the ASTUTE project (www.astute-project.eu), and by Innoviris (www.innoviris.be) through the VariBru project (www.varibru.be).

REFERENCES

1. N.B. Sarter. Multimodal information presentation: Design guidance and research challenges. *International journal of industrial ergonomics*, 36(5):439–445, 2006.
2. L.M. Reeves, J. Lai, J.A. Larson, S. Oviatt, TS Balaji, S. Buisine, P. Collings, P. Cohen, B. Kraal, J.C. Martin, et al. Guidelines for multimodal user interface design. *Communications of the ACM*, 47(1):57–59, 2004.
3. E. Tsiporkova, T. Tourwé, and N. González-Deleito. Towards a semantic modelling framework in support of multimodal interface design. *In Proceedings of the 13th IFIP TC13 Conference on Human-Computer Interaction, INTERACT 2011*, pages 636-639, 2011.
4. I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML., 2004.
5. Pellet: OWL 2 reasoner for Java.

---

[i] ASTUTE is a large EU project (www.astute-project.eu) which aims at defining a reference architecture for the development of human machine interactions, targeting proactive information retrieval and delivery based on the situational context, as well influenced by information content and services, and user state information. The ultimate goal is to design intelligent multi-modal interfaces enabling to determine which information and services to push to the user at the right time via the appropriate modality. The approach will be verified in several different industrial demonstrators in the domain of avionics, automotive and emergency management.