Output Multimodal Interaction: The Case of Augmented Surgery

Benoît Mansoux[†‡], Laurence Nigay[†] & Jocelyne Troccaz[‡]

[†] *CLIPS-IMAG / équipe IIHM, 385 rue de la Bibliothèque, 38041 Grenoble cedex 9, France*

Tel: *+33 4 76 51 43 65*

Email: *benoit.mansoux@imag.fr*

[‡] *TIMC-IMAG / équipe GMCAO, Faculté de Médecine, 38700 La Tronche, France*

Output multimodal interaction involves choice and combination of relevant interaction modalities to present information to the user. In this paper, we present a framework based on reusable software components for rapidly developing output multimodal interfaces by choosing and combining interaction modalities. Such an approach enables us to quickly explore several design alternatives as part of an iterative design process. Our approach is illustrated by examples from a computer-assisted surgery system that runs in a specific environment (i.e. an operating room) and so needs adapted multimodal interaction. Our approach supports the exploration of several output multimodal interaction design alternatives with the surgeons.

Keywords: multimodal presentation, software components, computer-assisted surgery systems.

1 Introduction

In this paper we focus on the software development of output multimodal interfaces (from the system to the user) by describing a component-based framework, called ICARE, which allows the easy and rapid development of multimodal interfaces. Our approach relies on our previous work: the ICARE framework for input multimodal interfaces [Bouchet et al. 2004]. In this paper we explain the extensions to the existing ICARE framework for the case of outputs.

Our goal is to define a framework to enable rapid development of output multimodal interfaces and therefore more iterations as part of an iterative user-centred design method for achieving usable multimodal user interfaces [Myers et al. 2000]. Our application domain is computer-assisted surgery requiring adapted multimodal interaction for a specific environment, the operating room. We are using our framework for cost-effectively exploring several output multimodal interaction design alternatives with surgeons.

The structure of the paper is as follows: first, we present related work on development frameworks and tools for multimodality. Second we present our extensions of the ICARE framework for output multimodal interaction by outlining the conceptual model that includes elementary and modality dependent components as well as generic components (reusable components) for combining modalities (fission mechanism) and its implementation. We finally illustrate the approach by considering the design of the output interface of a computer-assisted kidney puncture system, PERM.

2 Related Work: Tools for Multimodality

Although several multimodal systems have been built, their development still remains a difficult task. The existing frameworks dedicated to multimodal interaction are currently few and limited in scope.

Existing tools mainly focus on input multimodality, either by addressing a specific technical problem including the fusion mechanism [Flippo et al. 2003; Nigay & Coutaz 1995], the composition of several devices [Dragicevic & Fekete 2004] and mutual disambiguation [Oviatt 2000; Flippo et al. 2003], or by being dedicated to specific modalities such as gesture recognition [Westeyn et al. 2003], speech recognition [Glass et al. 2004] or the combined usage of speech and gesture [Krahnstoever et al. 2002]. Going one step further than providing a particular modality or generic reusable mechanisms (i.e. fusion and mutual disambiguation mechanisms), Quickset [Johnston et al. 1997] defines an overall implementation architecture as well as the Open Agent Architecture (OAA) [Moran et al. 1997]. Quickset mainly focuses on input multimodality based on speech and gesture and has been applied to the development of map-based systems.

For outputs, several studies have been performed in the context of the conversational paradigm, also called intelligent multimedia presentation in which seminal work is presented in [André et al. 1993]. The system is designed here as a partner for the user (computer-as-partner [Beaudoin-Lafon 2004]): an output communicative act as part of a natural dialogue between the user and the system is made perceivable by a multimodal presentation. Moreover the main focus of such existing output multimodal frameworks is to automatically generate the output presentation, also called presentation planning systems, based on a speech act, a context such as the current available interaction resources and a user's profile. For example in the Embassi demonstrator [Elting et al. 2003], the architecture is based on OAA and includes a dedicated agent to achieve the combination of output modalities. Based on a speech act, the current context and the user's profile, Embassi generates multimodal presentations that are rendered by a dynamic set of distributed agents.

Focusing on the direct manipulation paradigm (computer-as-tools [Beaudoin-Lafon 2004]), very few tools are dedicated to the design and development of output multimodal interfaces. MOST (Multimodal Output Specification Tool) [Rousseau et al. 2004] is a recent framework for multimodal output interaction which focuses on automatic generation of multimodal presentation based on the interaction context defined as the triplet <user, system, environment>. MOST includes a rule-based selection mechanism for generating the multimodal presentation. MOST therefore defines a reusable framework for developing adaptive multimodal systems and its focus is not on the design of multimodality but more on adaptability by providing an editor for specifying the adaptation rules. A more closely related tool to our ICARE framework is CrossWeaver [Sinha & Landay 2003]: it is a prototyping tool dedicated to non-programmer designers. The created prototypes may involve several input modalities and two output modalities: visual display and text-to-speech synthesis. CrossWeaver divides the design process into three steps. First, the designer makes various sketches to form a storyboard. She/he also decides which combinations of input and output modalities will be available for each sketch and for transitions between sketches. Then, the user tests the prototype with the available input and output modalities. Finally, thanks to a log of the user's actions in the previous step, the designer can analyse how multimodality is handled by the user, and can quickly change the combination of modalities to adapt the interaction. Implementation of CrossWeaver is also based on OAA. As opposed to CrossWeaver, our ICARE framework is a development tool that enables cost-effectively modifications of modalities and combinations of modalities as part of an iterative design process.

To sum up, in comparison with existing frameworks and tools, our ICARE framework is dedicated to output multimodal interaction enhancing the sensory-motor capabilities of an interface by enriching it with innovative output modalities, such as augmenting a surgical tool with a mini-screen. The computer is not a partner as in intelligent multimedia presentation but a tool (computer-as-tools [Beaudoin-Lafon 2004]) for enhancing the task of the user. Moreover our focus is on design exploration by providing a tool enabling rapid development of several output multimodal interaction design alternatives. We currently do not address the problem of automatic adaptation but more the one of adaptable output interfaces. The extent to which interaction techniques and modalities can be successfully selected automatically remains the subject of debate within the HCI research community [Chalmers & Galani 2004]. Moreover, for the case of augmented surgery, our application domain, automatic adaptation is not suitable, even adaptable interfaces must still be experimentally validated.

As defined by Myers et al. [2000], in the general context of user interface software tools, tools for multimodal interfaces must aim to have a low threshold (easy to use) while providing a high ceiling (how much can be done with the tool). Additionally, in order to take account of the ever-widening world of modalities, the tools must be easily extendable, an extensibility that we address in our ICARE framework by considering a component-based approach.

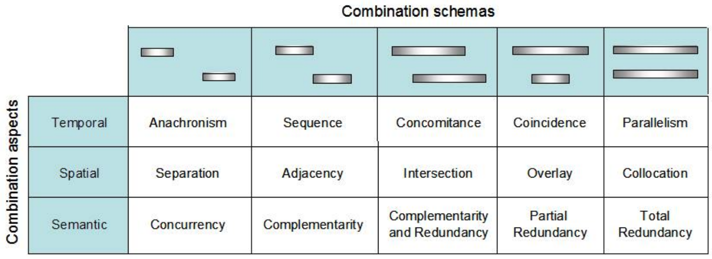


Figure 1: The combination schemas applied to two combination aspects, the temporal and spatial ones. Figure from Vernier & Nigay [2000].

3 Output Modality and Multimodality

We define an input (from the user to the system) / output (from the system to the user) interaction modality as the coupling of a device *d* with an interaction language $L :< d, L >$ [Nigay & Coutaz 1997]. For outputs (from the system to the user), a physical device delivers information. Examples of physical devices include loudspeakers and screens. An interaction language defines a set of well-formed expressions (i.e. assembly of symbols according to some conventions) that convey meaning. The generation of a symbol, or a set of symbols, involves actions on physical devices. Examples of interaction languages include pseudo-natural language and graphical animation. Our definition of an output modality enables us to extend the range of possibilities for output multimodality that implies multiple output modalities. Indeed a system can be multimodal without having several output devices. A system using the screen as the unique output device is multimodal whenever it employs several output interaction languages: indeed one device and multiple interaction languages raises the same design and engineering issues as using multiple modalities based on different devices. Our definition of output multimodality is therefore system-oriented and a user-centred perspective may lead to a different definition.

Moreover in the face of such an increasing variety of interaction modalities we can no longer expect to model each output modality in all their diversity at the concrete level. In order to reason about modalities at a higher level of abstraction, a core model must be defined for characterizing the modalities. Such a core model for modality integration will greatly help designers and programmers by allowing them to reason at a higher level of abstraction than the level of a particular modality. This is necessary to be able to select them for an efficient multimodal presentation. Towards this goal, a first set of properties has been proposed in [Vernier & Nigay 2000] for characterizing the interaction language that we reuse in our ICARE framework.

Although each modality can be used independently within a multimodal system, the availability of several modalities in a system naturally leads to the issue of their combined usage. The combined usage of multiple modalities opens a vastly augmented world of possibilities in user interface design. Our framework is based on the CARE properties [Nigay & Coutaz 1997] for reasoning about multimodal interaction: These properties are Complementarity, Assignment, Redundancy, and Equivalence that may occur between the modalities available in a multimodal user interface. We define these four notions (CARE) as relationships between devices and interaction languages and between interaction languages and tasks. Vernier & Nigay [2000] extends the CARE properties to further characterize the combination. Our resulting composition space is organized along two axes. The first axis ranges over a set of combination schemas, as presented in Figure 1.

These schemas use the five Allen [1983] relationships to provide a means of combining multiple modalities into a composite modality. The second axis considers five aspects for characterizing a combination: 1-Time, 2-Space, 3-Articulatory, 4-Syntactic and 5-Semantic. The most studied aspect of combination is the semantic one presented in Figure 1, where one considers the meaning of the conveyed information along the modalities (complementarity and redundancy). The articulatory (device) and syntactic (language) aspects of a combination are based on the definition of a modality as the coupling of a physical device d with an interaction language *L*. Finally the last remaining aspects, temporal and spatial, are presented in Figure 1. Temporal aspects of the combination have been studied in the literature and are related to the guiding principle in [Reeves et al. 2004]: 'to ensure system output modalities are well-synchronized temporally (for example map-based display and spoken directions)'. For spatial aspects, in the context of computer-assisted surgery systems, we have studied the spatial continuity in interaction [Dubois et al. 2002].

4 ICARE for Multimodal Output

Based on the definitions of the previous section, we here present the extensions to the existing ICARE framework for output multimodality. We first describe the new aspects of the ICARE conceptual model for output multimodality and then focus on its implementation.

4.1 ICARE Conceptual Model for Output Multimodality

The ICARE framework for input multimodality [Bouchet et al. 2004] is based on components and includes elementary and combination components. We reuse these two types of components for output multimodality.

Elementary components define building blocks useful for defining an output modality. The two types of elementary components are the Device and the Interaction Language components. An ICARE Interaction Language component communicates with a Device component via events, in order to form an output modality. Such elementary components are the same as for inputs except for the characteristics that describe them. Examples of characteristics for an output interaction language component include transient or sustained, precise or vague, local or global and deformed or not, as defined by Vernier & Nigay [2000]. In Figure 2, we present the elementary components of two modalities of a game prototype that we have developed using ICARE. The goal is to complete a physical puzzle and the system helps the player to correctly orient the puzzle pieces. For providing the guidance information, the output modalities are graphics displayed on a localized mini-screen fixed to the puzzle piece as well as voice messages.

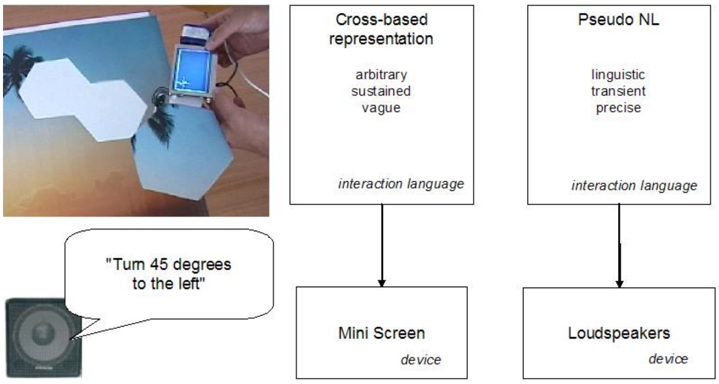


Figure 2: Examples of ICARE elementary components developed for a puzzle game. (a) A puzzle piece with a mini-screen that displays guidance information as two crosses, one mobile (the current orientation) and one static (the right orientation) (M1), while playing a pseudo natural language oral message (M2); (b) ICARE components for modality M1 = <cross-based graphical representation, mini-screen>; (c) ICARE components for modality M2 = <pseudo NL, loudspeakers>.

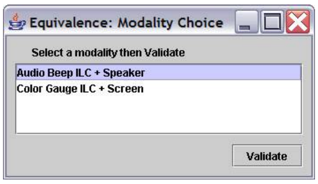


Figure 3: Modality selection window (part of a meta user interface).

Composition components are generic in the sense that they are not dependent on a particular output modality. Based on the CARE properties and the combination space (cf. Section 3), four combination components are defined. Such composition components for outputs are different than the ones defined for inputs: indeed while for multimodal inputs we classically define a fusion mechanism, for outputs one key design issue is a fission mechanism. Four composition components for outputs enable us to define a fission mechanism for a given presentation task.

The Redundancy component enables the parallel use of several equivalent modalities to present the same information. Redundant usage of modalities by forcing the user's perception reinforces the respect of two ergonomic criteria: observability and insistence. The Redundancy component receives an event and dispatches it to all the modalities linked to it. It corresponds to the case 'total redundancy' of Figure 1.

The Equivalence component, analogous to the Redundancy one, implies the use of equivalent modalities. Equivalence differs from Redundancy because the output modalities are not active at the same time. The Equivalence component receives an event and sends it to only one of its linked modalities. It implies a choice of modality done either by the user (adaptable system) or by the system (adaptive system). When done by the user, such a choice is specified by the user using input modalities as part of a meta user interface that enables the user to select the modalities amongst a set of equivalent modalities. Figure 3 presents a simple way of selecting an output modality by direct manipulation using a mouse. Multimodal input interaction can be defined for selecting the output modalities. As a conclusion our ICARE framework for inputs can be used for defining that meta user interface.

The Redundancy/Equivalence component mixes the Redundancy and Equivalence components behaviours. It corresponds to the Redundancy component where redundancy could be optional. This component allows the selection of one or more equivalent output modalities. In theory, this component is not necessary, but it makes the handling of equivalent modalities simpler. The Redundancy/Equivalence component receives an event and sends it to one or more modalities.

Finally, a Complementary component is used when a set of modalities is needed to convey information. Each modality carries a different piece of information. Complementary implies that the user combines the perceived data (fusion of perceived data) in order to interpret the conveyed information. But from a system point of view, the Complementary component performs data fission for output. The Complementarity component receives an event and sends a part of the information contained in this event to each modality. The application designer selects which part of information is sent to each modality. The complementary component implements the three cases ‘complementarity’, ‘complementarity and redundancy’ and ‘partial redundancy’ of Figure 1.

In Figure 4 we present an example of a Complementarity component as well as a Redundancy one for the game prototype of Figure 2. The ICARE diagram of Figure 4a describes a complementary use of two modalities, one for displaying the direction to turn the puzzle piece on the mini-screen while the exact angle is specified by an oral message. We could also decide using the same Complementarity component to display the direction and the angle on the mini-screen while repeating the angle by an oral message. In such a case, the Complementarity component is used for specifying a partial redundancy usage of the two modalities. The Redundancy component of Figure 4b implies a total redundancy: the guidance information is displayed on the mini-screen by two crosses while an oral message repeats the same guidance information in pseudo natural language way (i.e. ‘Turn 45 degrees to the left’).

4.2 ICARE Implementation Model for Output Multimodality

For implementing the ICARE components, since we extended our ICARE framework for multimodal input, we use the same component technology as for input: the JavaBeans technology. The properties of output modalities are class attributes which can be accessed/modified. The communication between ICARE components is based on the Java event model. To assemble two ICARE components, it is necessary that one component subscribes to events generated by the other component: we provide examples of subscribing in the following section.

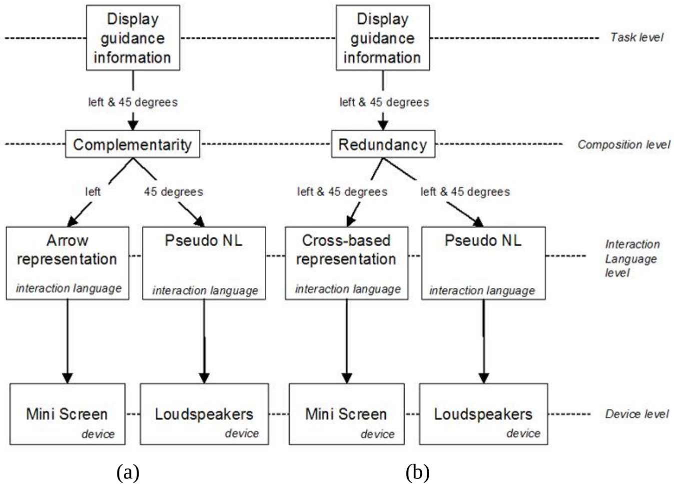


Figure 4: Examples of ICARE combination components developed for a puzzle game. (a) Two complementary modalities. (b) Two redundant modalities.

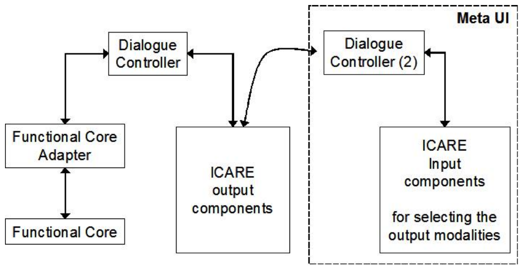
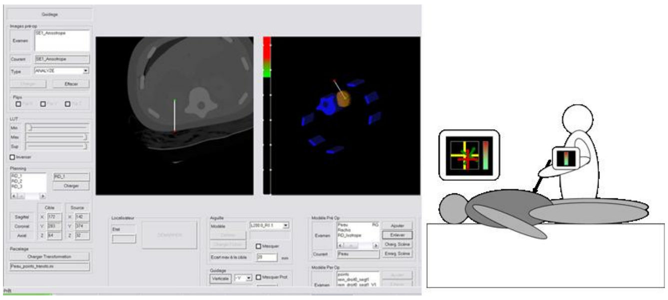


Figure 5: ICARE components within an ARCH software architecture and the meta user interface that enables the selection of equivalent modalities by the user.

As for input, the ICARE output components correspond to the two Interaction components of the ARCH software architectural model [UIMS 1992]. As shown in Figure 5, the Dialogue Controller defines the information to be presented (e.g. <turn 45 degrees left> in Figure 4) and corresponds to the task level. The ICARE components are then responsible for defining the multimodal presentation of the information. For the case of Equivalence and Redundancy/Equivalence components, a choice amongst the modalities must be performed as explained in the previous section (cf. Figure 3). If performed by the user, such a choice requires the definition of a meta user interface that includes a second Dialogue Controller (Dialogue Controller (2) in Figure 5) as well as ICARE input components for specifying the selection. The selection is then sent by the second Dialogue Controller to the ICARE output components.



(a) (b)

Figure 6: (a) The graphical interface of PERM, running on a PC. (b) The PERM setup with a desktop screen and a mini-screen.

So far, we have developed several ICARE output elementary components for the game prototype of Figure 2 as well as for the computer-assisted surgery system PERM that is described in the following section. For combination components, the four components described in Section 4.1 are developed. Moreover we recently started to address the temporal and spatial aspects of a combination, described in Figure 1. Control parameters must be added to the combination components, for example to specify that one modality is used first followed by the second one ('Sequence' case in Figure 1).

We currently manually assemble the ICARE output components as opposed to input ICARE components that are graphically assembled by direct manipulation in the ICARE graphical editor. When the output ICARE components will be inserted in the graphical editor, the developer/designer will graphically assemble the components without knowing the details of their implementations and from the resulting high level specification as in Figure 4, the code of the output multimodal UI will be then generated. So far we assemble the output components manually. We will explain the details of the manual assembling in the context of our PERM system.

5 Illustrative Example: The PERM System

We applied our ICARE approach for the development of the output interface of the Computer-assisted Surgery (CAS) system, PERM, a computer assisted kidney puncture, developed in collaboration with the Grenoble University Hospital. Our goal is to be able to quickly explore several design alternatives with the surgeon. PERM contains several phases corresponding to a predefined surgical protocol. It is a complex system and many parameters for different phases can be configured in the existing desktop user interface, represented in Figure 6a. We focus on the guiding task, which occurs during the surgical intervention. PERM assists the surgeon by providing in real time the position of the puncture needle according to a planned trajectory.

Since the desktop interface forces the surgeon to switch visual attention between the operating field and the guidance information displayed on screen, we decided to explore several design alternatives based on other output modalities such as sound and graphics on a mini-screen using our ICARE framework. While performing the puncture, few concepts are useful to the surgeon and include the real-time needle position and orientation and the planned trajectory. By using sound or by fixing a mini-screen onto the needle (Figure 6b) or on the surgeon's wrist, we can bring back important concepts within the operating field. A mini-screen is an innovative interaction device for CAS systems. As shown in Figure 6b, we can tie a mini-screen to the puncture needle. On the desktop screen, the whole set of guidance information (i.e. the needle position and orientation) is displayed. On the mini-screen, only the needle depth is displayed because it may be the most important piece of information at that time of the guiding task. Based on our design space for mini-screen organized along two dimensions, the usage of the mini-screen and the displayed information [Mansoux et al. 2005], various design solutions are defined. By developing the output interface with our ICARE framework, our goal is to quickly explore such design alternatives with a surgeon.

We have developed several ICARE elementary components for defining output modalities. Three output Device components are developed: the screen, the mini-screen and the microphone. We also developed several Interaction Language components: a colour gauge, a slider, a cross-based representation, a repeated sound in addition to the graphical presentation of the initial design that includes a 3D reconstruction and scanner images on top of which the performed trajectory is displayed (Figure 6a). We plan to develop more Interaction Language components before testing them with the surgeon. For example, a 2D colour gauge designed to fill the mini-screen will be developed. For exploring several design alternatives we will use our four composition components defined in Section 4.

In order to highlight the benefits of our approach (even though we are still assembling the components manually) we explain how we can easily change a modality in PERM and then how we can change a combination of modalities.

5.1 Changing a Modality

In PERM, once the needle is inserted into the patient's body from the planned entry point and with the right orientation, the surgeon must know how deep the needle is, according to the trajectory length. There are many ways to represent that ratio: needle depth / trajectory length. So far we developed three modalities. One modality is based on sound: a sound is repeated but the period of repetition is dynamic and based on the distance between the current location of the needle according to the target point. We adapt here the Doppler effect by varying the period of repetition instead of the amplitude of sound. The closer the needle is to the target point, the more frequently the sound is repeated (decreasing the period). Two other modalities are graphics displayed on the mini-screen tied to the needle. One graphical modality displays a colour gauge (Figure 7a) while the other one displays a slider (Figure 7b).

If we want to change the slider by the colour gauge, we need to replace one Interaction Language component by another and to connect the components again. The following few lines of code (pseudo Java) show how to create some components and to link them. Firstly, we create three components: one Device and two Interaction Languages.

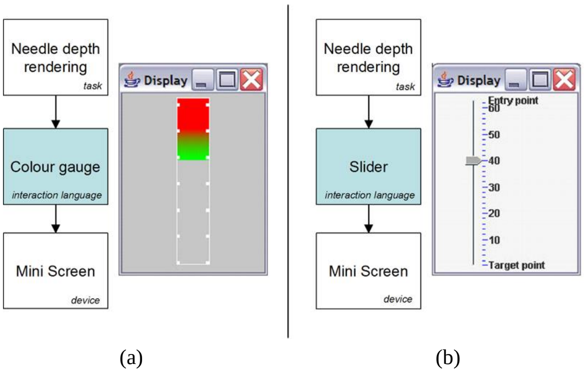


Figure 7: Two graphical modalities based on a mini-screen but with different interaction languages: (a) a colour gauge; (b) a slider.

```
MiniScreen myScreen = new MiniScreen ( . . . ) ;
ColourGauge aGauge = new ColourGauge ( . . . ) ;
Slider aSlider = new Slider ( . . . ) ;
```

Then we link two components: an Interaction Language with the Device.

```
aGauge.addListener ( myScreen ) ;
```

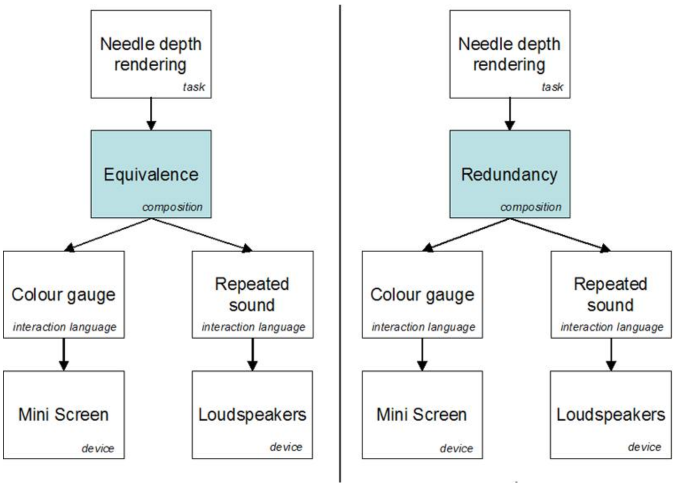
The MiniScreen is now listening to events coming from the ColourGauge. When the output components will be integrated in the graphical editor, the modification of a modality will be done graphically and the corresponding code will be automatically generated.

While changing a component in the ICARE assembling, it is possible that the developer needs to adjust the communication between the new component and the rest of the components. Adding extra code is sometimes needed to handle the new link. The following lines describe how to do it.

```
aSlider.addListener ( new ILListener ( ) {
  public void newData ( ICAREEvent e) {
    // non default behaviour
    // extra data processing added here
    . . .
    /* Create a new event with the transformed data. */
    ICAREEvent eNew = new ICAREEvent ( . . . ) ;
    myScreen.setData ( eNew ) ;
  }
} ) ;
```

In that specific case, the MiniScreen is not the listener of the interaction language any more. An anonymous Java class (of type ILListener) makes a bridge between the two components.

The example could appear as a simple change of graphical widgets because of the simple content carried by the modality. But an Interaction Language component conveys meaning and is able to adapt/transform the conveyed data. It is not limited to a simple widget and can be a more complex element such as a text-to-speech module.



(a) (b)

Figure 8: Switching from a confi guration with (a) Equivalence to one with (b) Redundancy.

5.2 Changing a Combination of Modalities

Changing a combination of modalities is straightforward and easier than changing a modality because the combination components are generic ones. Changing a combination of modalities involves simply switching one combination component by another one. Figure 8 shows the only change needed between an equivalent configuration and a redundant one for two modalities. Configuring the internal behaviour of the composition component (i.e. setting its parameters) is the only additional task that the designer may need to do.

Changing the composition components is straightforward and will enable us to quickly explore several design alternatives with the surgeon. For example by simply considering the two graphical modalities (slider and gauge) and the sound modality, several design solutions can be cost-effectively experimented with the surgeon.

Moreover we also plan to study the adaptation of the modalities during the surgical intervention (i.e. the meta user interface of Figure 5). For example, because the sound is less precise than a graphical representation, we can anticipate that the surgeon may need to change the modalities during the different phases of the intervention:

- only the sound is active when the needle is not touching the patient's body;
- the sound and the colour gauge are used redundantly when the needle is inserted; and
- only the colour gauge is active when the needle tip is very near the target point (e.g. < 10mm).

For this example, a Redundancy/Equivalence component will be used and linked to both modalities. Nevertheless making the output multimodal interface adaptable by the surgeon requires further studies in order to provide the adequate input modalities: we plan to explore voice commands and a pedal press for changing the output modalities.

Another key issue for adaptability, not applicable in Computer-assisted Surgery (CAS) systems since the surgeon is an expert of the system, is to make observable the available modalities and forms of multimodality. Finally we exclude automatic adaptation (adaptivity as opposed to adaptability) for CAS systems. For example let us consider that the automatic adaptation of the output modalities for presenting guidance information to a surgeon in PERM. Again the change of modalities is made at the turning point between two surgical phases: for example when the needle is not touching the patient's body, guidance information with anatomical and pre-operative information is displayed on a monitor and as soon as the needle is touching the patient's body, only the pre-planned trajectory is displayed as crosses on a mini-screen attached to the puncture needle. Although the surgeon is trained to use the system, such automatic adaptation may be a surprise. Moreover a very small backward movement of the needle may imply to come back to the presentation on the monitor, making the output interface very unstable (back and forth between the monitor and the mini-screen). Automatic adaptation in PERM is not planned and we will experimentally study adaptation by the surgeon using voice commands or a pedal press.

6 Conclusion

In this paper, we have presented the extensions to our ICARE framework for developing output multimodal interfaces. The approach is based on reusable software components for rapidly developing output multimodal interfaces by choosing and combining interaction modalities. We illustrated the framework by considering the development of several design alternatives of PERM, a computer-assisted kidney puncture system. Before further enriching the ICARE framework (including a graphical editor for assembling the output components and a mechanism based on psychological knowledge to guide the selection of the components), our current work is to test the design solutions of PERM with a surgical team. Our goal is three-fold:

1. evaluate the usability of the modalities and of the setting with the mini-screen;
2. test the adaptation of the output modalities by the surgeon during the surgical intervention; and
3. evaluate the approach itself as a tool to quickly explore design alternatives with the surgeon.

Acknowledgements

Special thanks to J. Bouchet and M. Serrano for their help in developing the ICARE components and to Mr. G. Serghiou for reviewing the paper. This work is partly funded by the SIMILAR European FP6 network of excellence dedicated to multimodality (http://www.similar.cc).

References

Allen, J. F. [1983], Maintaining Knowledge about Temporal Intervals, *Communications of the ACM* **26**(11), 832–43.

André, E., Finkler, W., Graf, W., Rist, T., Schauder, A., & Wahlster, W. [1993], WIP: The Automatic Synthesis of Multimodal Presentations, *in* M. T. Maybury (ed.), *Intelligent Multimedia Interface*, AAAI Press, pp.75–93.

Beaudoin-Lafon, M. [2004], Designing Interaction, Not Interfaces, *in* M. F. Costabile (ed.), *Proceedings of the Working Conference on Advanced Visual Interface (AVI 2004)*, ACM Press, pp.15–22.

Bouchet, J., Nigay, L. & Ganille, T. [2004], ICARE Software Components for Rapidly Developing Multimodal Interfaces, *in* R. Sharma, T. Darrell, M. Harper, G. Lazzari & M. Turk (eds.), *Proceedings of the 6th International Conference on Multimodal Interfaces (ICMI'04)*, ACM Press, pp.251–8.

Chalmers, M. & Galani, A. [2004], Seamful Interweaving: Heterogeneity in the Theory and Design of Interactive Systems, *in* D. Gruen & I. McAra-McWilliams (eds.), *Proceedings of the Symposium on Designing Interactive Systems: Processes, Practices, Methods and Techniques (DIS'04)*, ACM Press, pp.243–52.

Dragicevic, P. & Fekete, J.-D. [2004], The Input Configurator Toolkit: Towards High Input Adaptability in Interactive Applications, *in* M. F. Costabile (ed.), *Proceedings of the Working Conference on Advanced Visual Interface (AVI 2004)*, ACM Press, pp.244–7.

Dubois, E., Nigay, L. & Troccaz, J. [2002], Assessing Continuity and Compatibility in Augmented Reality Systems, *Universal Access in the Information Society* **4**(1), 263–73.

Elting, C., Rapp, S., Mölher, G. & Strube, M. [2003], Architecture and Implementation of Multimodal Plug and Play, *in* S. Oviatt, T. Darrell, M. Maybury & W. Wahlster (eds.), *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI'03)*, ACM Press, pp.93–100.

Flippo, F., Krebs, A. & Marsic, I. [2003], A Framework for Rapid Development of Multimodal Interfaces, *in* S. Oviatt, T. Darrell, M. Maybury & W. Wahlster (eds.), *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI'03)*, ACM Press, pp.109–16.

Glass, J., Weinstein, E., Cyphers, S., Polifroni, J., Chung, G. & Nakano, M. [2004], A Framework for Developing Conversational User Interfaces, *in* R. K. Jacob, Q. Limbourg & J. Vanderdonckt (eds.), *Proceedings of the 5th International Conference on Computer-Aided Design of User Interfaces (CADUI 2004)*, Kluwer, pp.347–58.

Johnston, M., Cohen, P. R., McGee, D., Oviatt, S. L., Pittman, J. A. & Smith, I. [1997], Unification-based Multimodal Integration, *in* P. R. Cohen & W. Wahlster (eds.), *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and of the 8th Conference of the European Chapter of the Association for Computational Linguistics*, Morgan-Kaufmann, pp.281–8.

Krahnstoever, N., Kettebekov, S., Yeasin, M. & Sharma, R. [2002], A Real-time Framework for Natural Multimodal Interaction with Large Screen Displays, *in* A. Waibel, W. Gao, E. Horvitz, S. Furui & J. Yang (eds.), *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces (ICMI'02)*, IEEE Computer Society Press, pp.349–54.

Mansoux, B., Nigay, L. & Troccaz, J. [2005], The Mini-Screen: An Innovative Device for Computer Assisted Surgery Systems, *in* J. D. Westwood, R. S. Haluck, H. M. Hoffman, G. T. Mogel, R. Phillips & R. A. R. anf K. G. Vosburgh (eds.), *Medicine Meets Virtual Reality 13: The magical Next Becomes the Medical Now*, Vol. 111 of *Studies in Health Technology and Informatics*, IOS Press, pp.314–20.

Moran, D. B., Cheyer, A. J., E., J. L., Martin, D. L. & Park, S. [1997], Multimodal User Interfaces in the Open Agent Architecture, *in* A. R. Puerta & E. Edmonds (eds.), *Proceedings of the 2nd International Conference on Intelligent User Interfaces (IUI'97)*, ACM Press, pp.61–8.

Myers, B., Hudson, S. E. & Pausch, R. [2000], Past, Present, and Future of User Interface Software Tools, *ACM Transactions on Computer–Human Interaction* **7**(1), 3–28.

Nigay, L. & Coutaz, J. [1995], A Generic Platform for Addressing the Multimodal Challenge, *in* I. Katz, R. Mack, L. Marks, M. B. Rosson & J. Nielsen (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, ACM Press, pp.98–105.

Nigay, L. & Coutaz, J. [1997], Multifeature Systems: The CARE Properties and Their Impact on Software Design, *in* J. Lee (ed.), *Intelligence and Multimodality in Multimedia Interfaces: Research and Applications*, AAAI Press, Chapter 9. http://iihm.imag.fr/publs/1995/IMMIChapNigay95.pdf, last accessed 2006-05-27.

Oviatt, S. L. [2000], Taming Recognition Errors with a Multimodal Interface, *Communications of the ACM* **43**(9), 45–51. Special Issue on Spoken Language Interfaces.

Reeves, L. M., Lai, J., Larson, J. A., Oviatt, S., Balaji, T. S., Buisine, S., Collings, P., Cohen, P., Kraal, B., Martin, J. C., McTear, M., Raman, T. V., Stanney, K. M., Su, H. & Wang, Q. Y. [2004], Guidelines for Multimodal User Interfaces Design, *Communications of the ACM* **47**(1), 57–9.

Rousseau, C., Bellik, Y., Vernier, F. & Bazalgette, D. [2004], Architecture Framework For Output Multimodal Systems Design, *in Proceedings of Australian Conference on Computer–Human Interaction (OzCHI 2004)*, CHISIG. http://www.limsi.fr/Individu/vernier/Publis/OZCHI04-LongPaper.pdf last accessed 2006-05-27.

Sinha, A. K. & Landay, J. A. [2003], Capturing User Tests in a Multimodal, Multidevice Informal Prototyping Tool, *in* S. Oviatt, T. Darrell, M. Maybury & W. Wahlster (eds.), *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI'03)*, ACM Press, pp.117–24.

UIMS [1992], A Metamodel for the Runtime Architecture of an Interactive System: The UIMS Tool Developers Workshop, *ACM SIGCHI Bulletin* **24**(1), 32–7.

Vernier, F. & Nigay, L. [2000], A Framework for the Combination and Characterization of Output Modalities, *in* P. Palanque & F. Paternò (eds.), *Interactive Systems. Design, Specification, and Verification: Proceedings of the 7th International Workshop, DSV-IS 2000*, Vol. 1946 of *Lecture Notes in Computer Science*, Springer-Verlag, pp.35–50.

Westeyn, T., Brashear, H., Atrash, A. & Starner, T. [2003], Georgia Tech Gesture Toolkit: Supporting Experiments in Gesture Recognition, *in* S. Oviatt, T. Darrell, M. Maybury & W. Wahlster (eds.), *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI'03)*, ACM Press, pp.85–92.

Author Index

Keyword Index