Temporal Aspects of CARE-based Multimodal Fusion: From a Fusion Mechanism to Composition Components and WoZ Components

Marcos Serrano and Laurence Nigay
University of Grenoble, CNRS, LIG
B.P. 53, 38041, Grenoble Cedex 9 France
{Marcos.Serrano, Laurence.Nigay}@imag.fr

ABSTRACT

The CARE properties (Complementarity, Assignment, Redundancy and Equivalence) define various forms that multimodal input interaction can take. While Equivalence and Assignment express the availability and respective absence of choice between multiple input modalities for performing a given task, Complementarity and Redundancy describe relationships between modalities and require fusion mechanisms. In this paper we present a summary of the works we have carried using the CARE properties for conceiving and implementing multimodal interaction, as well as a new approach using WoZ components. We present different technical solutions for implementing the Complementarity and Redundancy of modalities with a focus on the temporal aspects of the fusion. Starting from a monolithic fusion mechanism, we then explain our component-based approach and the composition components (i.e., Redundancy and Complementarity components). As a new contribution for exploring design solutions before implementing an adequate fusion mechanism as well as for tuning the temporal aspects of the performed fusion, we introduce Wizard of Oz (WoZ) fusion components. We illustrate the composition components as well as the implemented tools exploiting them using several multimodal systems including a multimodal slide viewer and a multimodal map navigator.

Categories and Subject Descriptors

H.5.2 **[Information Interfaces and Presentation]**: User Interfaces– Input devices and strategies, Interaction styles, Prototyping, User interface management systems (UIMS); D.2.2 **[Software Engineering]**: Design Tools and Techniques – User interfaces

General Terms

Algorithms, Human Factors, Theory.

ICMI'09, November 2–6, 2009, Cambridge, MA, USA.

Keywords

Multimodal Interaction, Fusion, Component-based Approach, Design and Implementation Tool, Wizard of Oz.

1. INTRODUCTION

Several research projects have demonstrated very innovative modalities as part of prototypes and we are facing a huge range of possibilities in terms of interaction modalities. In this context, there is a specific need for generic models and tools allowing us to do fast prototyping of multimodal applications. These generic tools should enable the rapid development of multimodal prototypes to be experimentally tested as part of an iterative user-centered process.

One of the key points concerning those generic tools is the fusion mechanism. Through the fusion of input modalities, new combined modalities are defined (i.e., pure and combined interaction modalities are introduced in [2]), enriching the design space for interaction. As explained in [19], fusion is classically classified as early fusion or late fusion. Early fusion represents fusion at the signal level, and late fusion represents fusion at the semantic level. Furthermore, semantic fusion can be divided into subprocesses. In [12], three subprocesses for multimodal fusion are identified: coordination, content fusion and event fusion. The coordination subprocess is in charge of creating pairs of coordinated events coming from two input modalities. Content fusion obtains a coherent sense from incomplete information. Event fusion produces a resulting complex act from pragmatic monomodal acts. Content and event fusion subprocesses are closely related to the application, while the coordination one is related to temporal fusion. In this paper we focus on those temporal aspects of generic fusion mechanisms.

For studying the fusion, we based our works on the CARE properties. The CARE properties (Complementarity, Assignment, Redundancy and Equivalence) define various forms that multimodal interaction can take. While Equivalence and Assignment express the availability and respective absence of choice between multiple modalities for performing a given task, Complementarity and Redundancy describe relationships between modalities and require fusion mechanisms. Since our first definition [8], the CARE properties have been used to implement several multimodal applications.

In this paper we present a summary of the different technical solutions for multimodal interaction that have been defined based on the CARE properties and we illustrate them by considering several examples of multimodal applications. Those solutions have evolved from a monolithic algorithm integrated in the PAC-Amodeus software architecture to the implementation of several software components dedicated to each CARE property. We also introduce a new contribution based on the use of the CARE properties and Wizard of Oz (WoZ) components.

For presenting our work on the CARE-based fusion mechanisms, this paper is organized as follows. We first explain in detail the combination space defined by the CARE properties and the temporal relationships between input modalities. We then describe the different approaches that have been taken for implementing multimodal interaction based on the CARE properties, from a single fusion mechanism to several composition components. Among the composition components, we distinguish operational components and WoZ components.

2. COMBINATION SPACE

The combination space of interaction modalities is organized along two dimensions. The first dimension defines the type of relationships between modalities based on the CARE properties. The second dimension is related to the temporal relationships between input modalities.

2.1 CARE properties

The CARE properties [8] were proposed as a simple way of characterizing and assessing aspects of multimodal interaction: the Complementarity, Assignment, Redundancy, and Equivalence that may occur between the interaction modalities available in a multimodal user interface. They have been shown to be useful concepts for the design and evaluation of multimodal interfaces in [8]. In this paper, we focus on the technical solutions for implementing the CARE properties.

The formal expressions of the CARE properties rely on the notions of *state*, *goal*, *modality* and *temporal relationships*. A *state (s)* is a set of properties that can be measured at a particular time. A *goal (g)* is a state that an agent intends to reach. An *agent* is an entity (user, system or component) capable of performing actions. A *modality (m)* is an interaction method that an agent can use to reach a goal. A sequence of successive steps is called an *interaction trajectory*. Two examples of a modality can be the general terms 'using speech' or 'using microphone'. A *temporal relationship (TR)* characterizes the use over time of a set of modalities. The use of these modalities may occur simultaneously or in sequence within a *temporal window (TW)*, that is, a time interval. Figure 1 illustrates these notions, used for expressing the CARE properties.

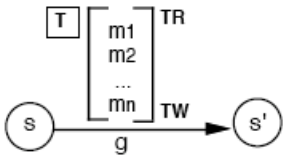

Figure 1. Notation for expressing the CARE properties [8].

From those concepts, the CARE properties are defined as a set of properties that characterize four types of relationships between modalities for reaching state s' from state s. The letter T in Figure 1 represents the CARE properties in the diagram**.**

Equivalence expresses the availability of choice between multiple modalities but does not impose any form of temporal constraint on them. More formally, modalities of set M are *equivalent* for reaching s' from s, if it is necessary and sufficient to use any one of the modalities.

In contrast to equivalence, **assignment** expresses the absence of choice. More formally, modality m is *assigned* in state s to reach s', if no other modality can be used to reach s' from s.

Two modalities are used **redundantly** to reach state s' from state s, if they have the same expressive power (they are equivalent) and if they are used within the same temporal window. In other words, the two modalities are required to reach state s' if they are used redundantly, and they convey the same meaning.

Modalities of a set M are used in a **complementary** way to reach state s' from state s within a temporal window, if all of them must be used to reach s' from s, i.e., none of them taken individually can cover the target state.

As opposed to Equivalence and Assignment, Redundancy and Complementarity imply fusion of input modalities. The formal expressions of the CARE properties include the notion of temporal relationship *(TR*) that we further refine by the second dimension of our combination space.

2.2 Temporal relationships

We define the temporal relationships between two modalities by using Allen's properties [1]. We used Allen properties to define the different combination schemas between two modalities related to the combination aspects (temporal, spatial, syntaxic, semantic) in [23]. In this paper we focus on the temporal aspect of the fusion mechanisms. Figure 2 presents the different combination schemas for the temporal aspect. Each bar represents the temporal length of a modality (considering a left–to-right time axis).

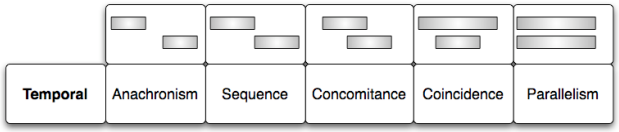

Figure 2. Allen's temporal properties.

Two modalities are combined anachronously if there is a temporal gap between their usage. Anachronism and sequence are distinguished by the size of the temporal window between the usage of the two modalities.

Two modalities are concomitant when one modality overlaps another one with a time interval during which the two modalities coexist.

Finally, the coincidence of two modalities is when one modality is used in the temporal context of another one. Such a combination is necessary to implement a modality that can only be used with another one.

A concomitant, coincident as well as a parallel combination implies that the two devices corresponding to the two modalities can function in parallel. These three temporal relationships refine the notion of simultaneous usage of modalities introduced in the previous section (TR) while anachronism and sequence further refine the sequential usage of modalities (TR).

2.3 Multimodal fusion examples

The CARE properties along with the temporal relationships have been shown to be useful for multimodal **input** interaction as well as for multimodal **output** interaction.

A first example of multimodal **input** interaction using the previous combination space can be found in [8], the MATIS system. MATIS stands for Multimodal Air Travel Information System. MATIS allows a user to retrieve information about flight schedules using speech, direct manipulation, keyboard and mouse, or a combination of these methods, supporting individual and combined use of multiple input modalities. For example, using a single modality, the user can say "show me the flights from Boston to Pittsburgh", can type sentences in pseudo-natural language in a dedicated text window, or can fill in a form using the keyboard and the mouse. When exploiting combined usage of modalities, the user may also combine speech and gesture as in "show me the flights from Boston to this city" along with the selection of "Pittsburgh" with the mouse on the screen ("put that there" paradigm [3]).

Although we focus on input multimodality in this paper, we present an example of **output** multimodality highlighting the fact that our combination space can also be applied to output multimodality. We consider a multimodal application that we developed with only one output device (i.e. screen). This example consists of a visualization system for large tables of numerical data, the MulTab system [23]. The system uses several output modalities based on the same output device, the screen. There are three different modalities: M1 displays the entire table. M2 is used to color each cell of the table according to the numerical data. M3 displays a part of the table deformed. Those three modalities can be combined in different ways. In Figure 3, M1 and M3 are combined in a Complementary way (CARE dimension, Section 2.1), and the temporal aspect of the combination is parallel (Temporal dimension, Section 2.2).

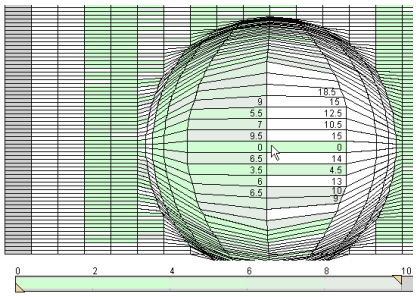

Figure 3. Combination of two output graphical modalities in the MulTab system (from [23]): a case of complementarity-parallelism.

After defining our CARE combination space and illustrating it, we now present the different implementations of it, from a monolithic fusion algorithm to composition components.

3. CARE: FUSION ALGORITHM IN PAC-AMODEUS

The CARE properties were first implemented as a fusion engine in the PAC-Amodeus software architecture [14]. The PAC-Amodeus model along with the fusion engine formed a reusable global platform applicable to the software design and implementation of multimodal applications.

PAC-Amodeus is a conceptual model that blends together the principles of both Arch [22] and PAC [7]. PAC-Amodeus has been used to implement the MATIS system, presented in the previous section.

The fusion mechanism relies on a uniform representation: a melting-pot which is a 2D structure. On the vertical axis, the "structural parts" model the composition of the task objects. For example, destination and time departure are the structural parts of the task objects handled for MATIS. The horizontal axis represents the time. Fusion is performed on those melting pots, as shown in Figure 4. There are three types of implemented fusion in PAC-Amodeus: microtemporal fusion, macrotemporal fusion, and contextual fusion. Microtemporal fusion is used to combine input events produced in parallel or in a pseudo-parallel manner (i.e., Parallelism, Coincidence or Concomitance along the temporal dimension of our combination space of Figure 2). Macrotemporal fusion is used to combine input events produced sequentially (i.e., Anachronism or Sequence in Figure 2). Contextual fusion is used to combine related input events produced without attention for temporal constraints.

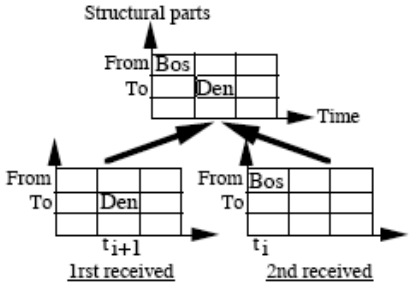

Figure 4. Macrotemporal fusion of two melting pots in PAC-Amodeus (from [7]).

While this fusion mechanism defines a reusable software solution implementing the redundancy and complementarity of input modalities, we also study the development of a fusion mechanism by adopting a component-based approach. Indeed such an approach offers the established advantages of verifying the software engineering properties of reusability maintainability and evolution [21]. The next section presents our general component-based model for multimodal interaction as well as the composition components based on CARE.

4. CARE-BASED COMPOSITION COMPONENTS

Before focusing on the composition components for performing fusion, we present our underlying component-based approach for multimodal interaction.

4.1 Component-based approach for multimodal interaction

Our approach [20] defines a set of characteristics of components, understanding the term "component" as any type of software unit (e.g., software component, service). It does not define the execution behavior of the components as do traditional component-based models, such as the Corba Component Model (CCM) [15] or JavaBeans [10]. The aim of this approach is to be used to define and implement high-level characteristics of components that can be implemented in any of those technologies.

Our characterization space of software components for multimodal interaction is constructed along three dimensions [20]. In the present work on fusion, we use two of those three dimensions (Figure 5). The **first** dimension is related to the **data-flow** from input devices to an interactive application. Along this axis, there are four types of components: Device, Transformation, Composition and Task. Fusion is performed at the Composition level.

The **second** dimension describes the **genericity** of the components. The space includes both generic and tailored components. Generic components represent high-level reusable abstractions. Tailored components implement operations for specific devices/modalities or for application-dependent tasks. It is possible to integrate existing tailored composition components in our approach, such as the fusion mechanism presented in [13], which combines speech and pointing. This fusion mechanism is tailored because it is an integrative recognition method based on a set of possible associations between an utterance and a gesture in the application context of explanations of geometry problems (i.e., modality and task dependent). However, in this paper we only present generic composition components, defining elementary temporal fusion operations, independent from modalities and application tasks. The Composition components are related to Redundancy and Complementarity of CARE.

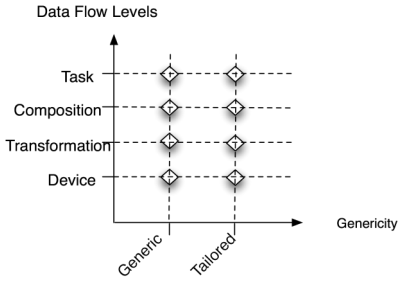

Figure 5. Two dimensions of our component-based approach [20].

4.2 Generic composition components

We have developed two implementations of the Composition components, first in ICARE [5] and more recently in OpenInterface [16].

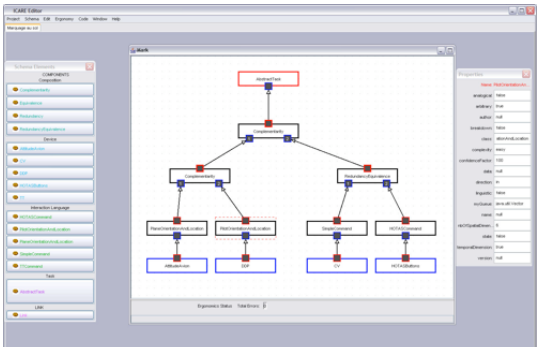

Figure 6. ICARE: Graphical assembling of components.

Those two software tools are based on a similar component-based approach that consists of visually assembling components in a graphical editor in order to define the pipeline corresponding to the abstracting function from actions using devices to tasks. The JavaBeans technology [10] was chosen to develop ICARE software components. As opposed to ICARE, the OpenInterface (OI) framework [16] is independent of the component technology and is able to handle distributed heterogeneous components based on different technologies (Java, C++, Matlab, Python, .NET). Figure 6 illustrates the graphical editor of ICARE and Figure 7 the graphical editor of OpenInterface, namely the OIDE (OpenInterface Interaction Development Environment). The OIDE allows the designer to define a multimodal interaction by assembling components. This assembly of components can be directly executed (no code is generated): each component will run separately, sending and receiving events from other components.

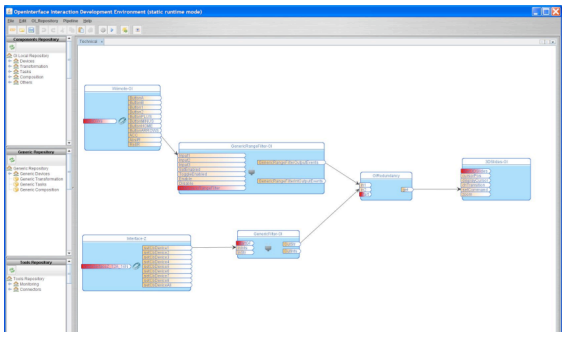

Figure 7. OpenInterface: Redundant composition of modalities graphically defined using the OIDE.

We have implemented generic composition components within those two frameworks corresponding to Redundancy and Complementarity of CARE. Composition components implemented in ICARE are described in [5]. They were used for developing several multimodal systems, such as the FACET system (plane cockpit simulator [5]), MEMO, a GeoNote system [4] or MID, a Multimodal Identification system [4] that enables the user to identify herself/himself through three modalities.

In this paper we focus on the two composition components implemented in OpenInterface. Those composition components have two inputs, one for each input modality, and one output, for sending the resulting combined event. In the case of Redundancy, when the component receives two events within the same temporal window, it sends one of them (first port event by default). As the two events contain redundant information, there is no need to send both events. The Complementarity component waits for receiving two events within the same temporal window and sends a concatenation of both events. The Redundancy and the Complementarity components have a configuration port that is used to define the size of the temporal window. This temporal window is defined by the designer/developer using the OIDE and before executing the component assembly.

Based on these two composition components, we have implemented several multimodal applications using the OpenInterface framework, such as a multimodal map navigator [20] and a multimodal arkanoid game. We use a multimodal slide viewer application to illustrate the composition components. This application allows the user to navigate within a list of slides using a combination of gestures. Two devices are used for the interaction: the wiimote and the "wiisotf" (Figure 8). The ‘wiisoft’ is a torsion sensor inside a wiimote plastic cover (InterfaceZ sensor [11]). The pressure on the "wiisoft" is used to change the slide or to zoom. The wiimote is used to change the slide by rotating the device or to point on the current slide.

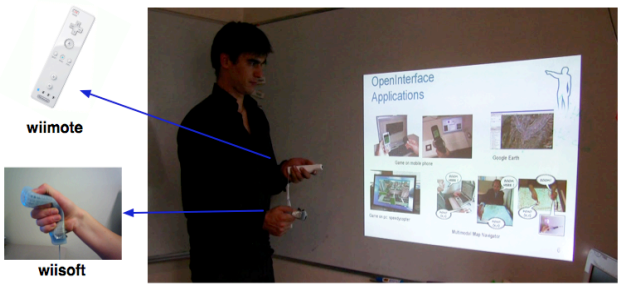

Figure 8. Multimodal slide viewer using the 'wiisoft' and the wiimote.

In an equivalent combination of the modalities, which does not imply fusion and therefore composition components, the user can move to the next slide by rotating the wiimote to the right or by pushing the "wiisoft".

In a redundant combination of modalities, the user can move to the next slide by rotating the wiimote and by pushing the "wiisoft" at the same time. The user must perform the two gestures at the same time (simultaneously within a temporal window TW) in order to accomplish the task. Figure 9 illustrates this redundant combination using the notation presented in Section 2.

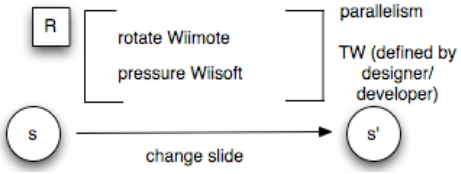

Figure 9. Redundancy of Wiimote and Wiisoft expressed with the CARE notation.

Figure 10 shows the corresponding component assembly of the redundant combination of the two modalities. Such assembly of components is graphically specified in the OIDE (the component assembly of Figure 7).

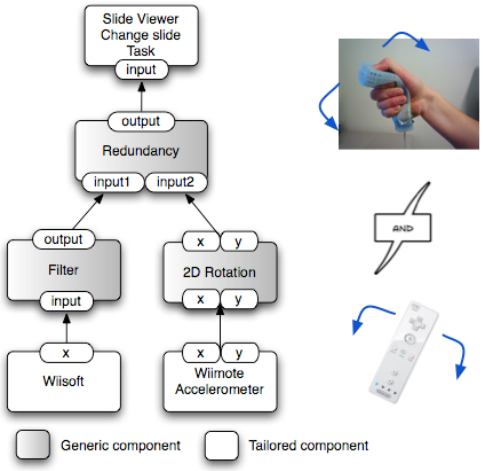

Figure 10. Redundancy component for combining two input modalities.

In a complementary combination of modalities, the user can zoom on a point of the current slide by pointing with the wiimote and pressing the 'wiisoft'. The wiimote will define the point to be zoomed while the 'wiisoft' will define the zoom value. Figure 11 illustrates the complementarity combination using the notation presented in Section 2.

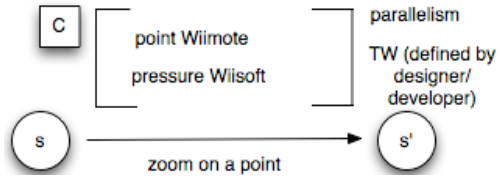

Figure 11. Complementarity of Wiimote and Wiisoft expressed with the CARE notation.

Figure 12 shows the component assembly of the complementarity combination of these two modalities. We can note two differences from the previous assembly (Figure 10): the 2D Position component and the composition component (reusability of the component assembly). Such modifications can therefore be graphically done quickly using the OIDE (Figure 7) in order to explore different multimodal design solutions.

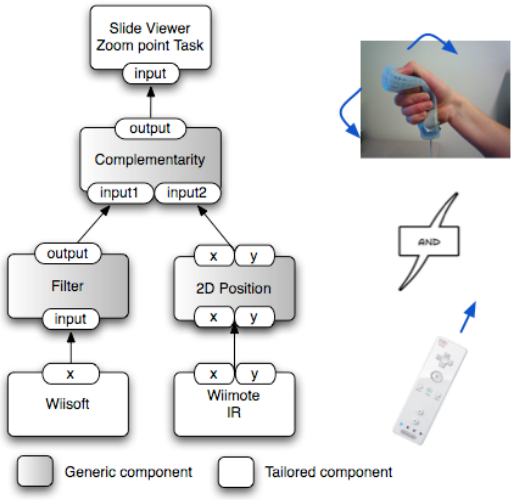

Figure 12. Complementary component for combining two input modalities.

Within an assembly of components, several Composition components can also be used to define further complex combinations, involving several types of fusion. For example, let us consider a new modality based on a pedal. In order to reduce recognition errors from the 'wiisoft' sensor, which is too sensible and launches erroneous zoom actions, we add a redundant composition of the 'wiisoft' and the pedal. The user has to push the pedal and the 'wiisoft' at the same time in order to be able to specify the zoom action (i.e, push-to-gesture similar to push-to-talk in the case of speech recognition). Figure 13 illustrates this example using two Composition components, the new added Redundancy component and the Complementarity component as in Figure 12.

We have presented Composition components for performing fusion as part of a model-based approach for prototyping multimodal interaction. Based on the CARE properties, the two composition components are generic and reusable for different input modalities. For the fine tuning of the temporal window of these composition components, we have also developed a novel approach: Wizard of Oz (WoZ) composition components.

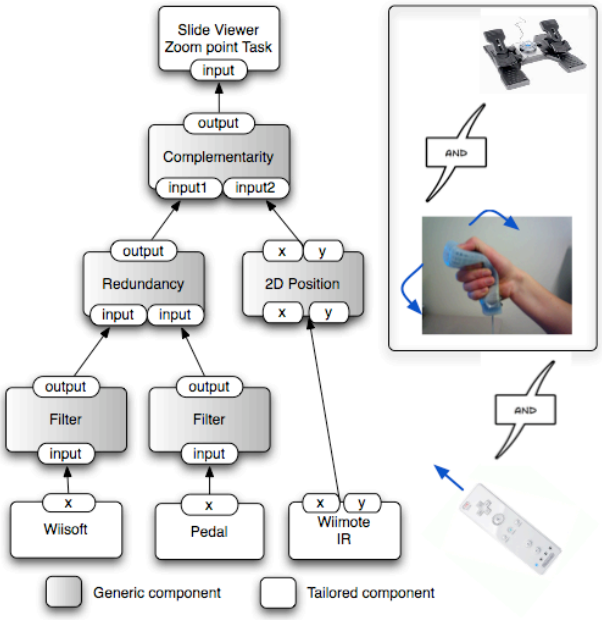

Figure 13. Complementarity and Redundancy for combining three input modalities.

4.3 WOZ composition components

As explained in [9], "a Wizard of Oz prototype is an incomplete system that a designer can simulate 'behind curtain' while observing the reactions of real end users". As pointed out in [6], WoZ studies have been shown to allow fast prototyping of interactive applications by allowing a human operator (i.e., a wizard) to simulate missing functions.

For the case of the Composition components, the Redundancy and Complementarity components presented before, it can be relevant to allow a human operator (i.e., a wizard) to decide in real time which type of temporal relation is used. Indeed, as pointed out in [18], individual differences appear between users in their multimodal integration pattern. Analyses revealed that everyone has a dominant integration pattern, either simultaneous or sequential, which is consistent and remains stable over time. In order to quickly identify a person's dominant integration pattern in real time and to adapt the system, we use a Wizard of Oz (WoZ) approach. A WoZ approach for adapting the system to user's dominant pattern was implemented in [17]. Our approach generalizes that work by proposing a component-based WoZ approach, which offers the benefit of being reusable and generic. We first introduce our WoZ component-based approach for multimodal interaction before presenting the WoZ Composition components.

4.3.1 WoZ components for multimodal interaction

We define WoZ components as parts of our component-based approach described in Section 4.1. These WoZ (non-functional) components are characterized according to the roles that a WoZ component can play in the data-flow of input multimodal interaction, from devices to tasks (vertical axis in Figure 5).

This approach allows building different types of prototypes with different degrees of functionality (according to the number of functional/non-functional components). This enables testing and improving functional components while evaluating the overall multimodal interaction. During the evaluation, the wizard can identify errors in those functional components that can be fixed later. Moreover, at each iteration of the design cycle, the designer/developer can replace some of the non-functional components by functional components, towards a final functional prototype that better fits the users' needs and preferences.

All generic WoZ components have the same interface configuration. Such a graphical user interface (GUI) is manipulated by the wizard. The WoZ component GUI is composed of several panels. Figure 14 shows the general GUI configuration of a WoZ component corresponding to a one-dimension device. This WoZ component allows the wizard to generate events corresponding to a one-dimension device (e.g., rotatory pot, button). Our experiences show that simulating a device is possible when the data generated by the device is simple. When dealing with complex input (images, sound), the wizard will use WoZ command components, which allow generating data at a higher level of abstraction. At the bottom of the window, two console panels display input and output events. Input events correspond to the events being received by the component. Some WoZ components may not have any input data. Output events correspond to the events being generated by the wizard. On the top left, a panel is dedicated to generating WoZ events, in this case with a slide bar. On the top right, a panel contains the configuration elements. Those two top panels are different for each type of WoZ components. The GUI of the WoZ components does not contain the video frame where the wizard will observe the end-user in action. This video frame is displayed on a dedicated terminal next to the computer, as it is usually done in evaluation studios.

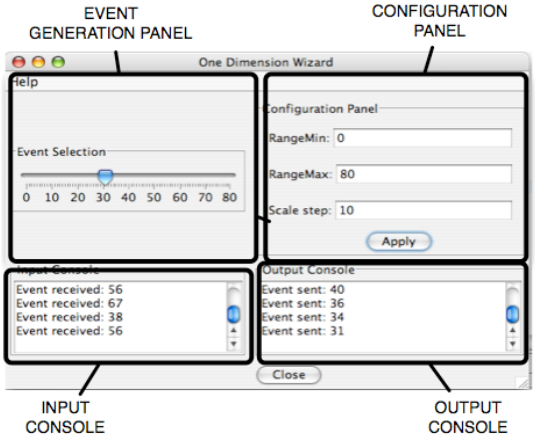

Figure 14. General User Interface of WoZ components.

Two main solutions have been taken in order to reduce the work load of the wizard in case there are several WoZ components in the assembly of components. Firstly, we consider multi-wizard configurations. In this case, the GUI of the WoZ components are displayed and controlled by different wizards on different computers (Figure 15).

Secondly, in the case of a single wizard, all the GUI of the WoZ components being displayed on the same computer are integrated into a single GUI. In the following section, we illustrate this point in the case of WoZ Composition components.

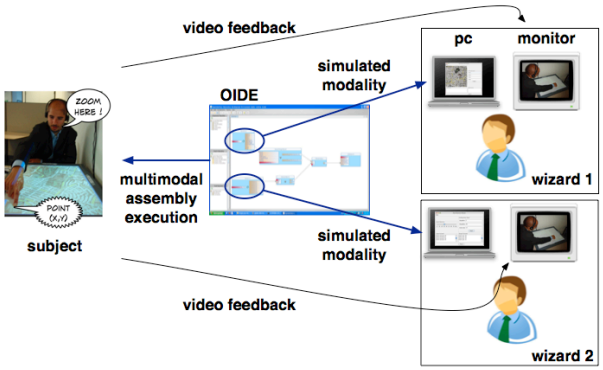

Figure 15. Multi-wizard configuration.

4.3.2 WoZ Composition components

In order to adapt the temporal fusion to the user's dominant integration pattern, the designer or developer can use a WoZ Composition component. WoZ Composition components also allow replacing of a missing input modality, while still using the events coming from the other one. Based on the received input events as well as the actions of the wizard, the Composition components generate the outputs.

For illustrating WoZ Composition components, we use the same zoom task of the slide viewer example (Figure 12), but this time for the case of a multimodal map navigator. It consists of a map that can be controlled using several interaction modalities in order to perform a set of interactive tasks such as panning or zooming. For example, the zoom on a specific point of the map can be specified by combining speech and pointing gesture (Figure 16). Capture of the finger position action can be done using several innovative sensors. Figure 16 shows the use of the DiamondTouch tactile surface to capture the finger position.

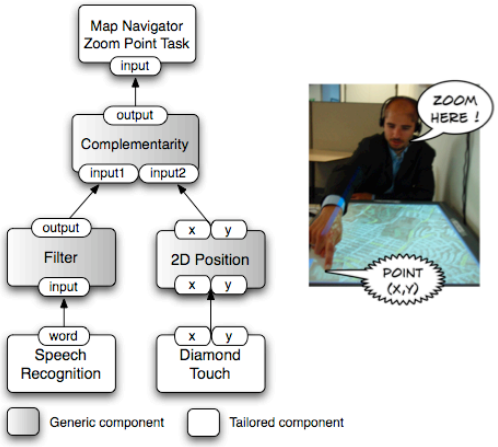

Figure 16. Illustrative example of multimodal interaction on a map and the corresponding component assembly.

Figure 17 illustrates the use of a WoZ Composition component along with a WoZ command component, which replaces the speech command recognition. The WoZ Command component is for example useful to perform experimental tests without having biased results due to a poor recognition accuracy of speech commands. For the case of several WoZ components and one wizard, the Composition component imports GUI elements from the other WoZ components: in Figure 17, we can see that the GUI of the WoZ Complementarity component includes elements of the generation and the configuration panels (i.e., buttons, configuration text labels) of the WoZ Command component.

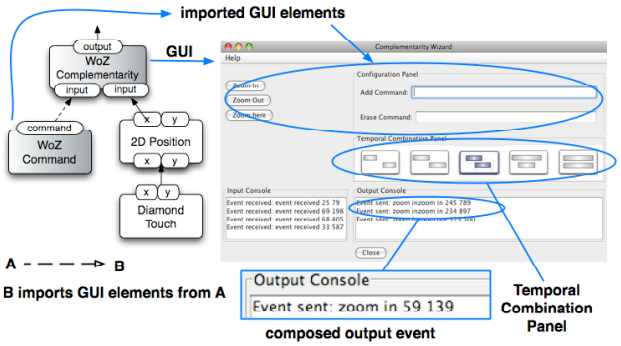

Figure 17. Graphical interface of the WoZ Composition component in the case of a partially simulated multimodal prototype.

In this example, the wizard observes the user and generates commands such as <zoom in> or <zoom out> according to the user's spoken utterances. The WoZ Composition component generates a combined output event by coordinating the command event (launched by the wizard using one of the command buttons) and the input event coming from the DiamondTouch. In order to define the temporal combination of the two input events and to modify it in real time according to user's dominant integration pattern, we use the temporal relationships presented in Figure 2. The temporal relationships are displayed in the GUI (Figure 17). The wizard can select a temporal combination in the list and define its time value by double-clicking on it (Figure 17). This will define which event of the DiamondTouch is used to generate the combined event, when the wizard clicks on a command button.

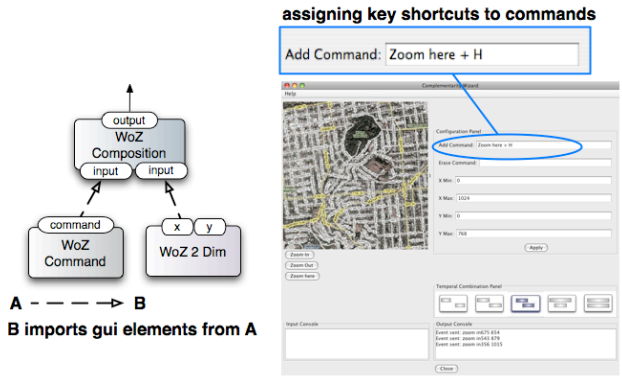

Figure 18. Graphical interface of the WoZ Composition component in the case of a fully simulated prototype.

In case the designer or the developer wants to fully simulate a multimodal interaction, they can create a non-functional prototype using WoZ components. For instance, in our running example, we want to simulate the tactile surface in order to allow a vertical setting (map projection on a wall) since we currently have no component for capturing finger pointing on a wall. We connect two WoZ components, one for simulating the speech commands and another one for simulating the 2D finger pointing, to a WoZ Composition component. As seen previously, this Composition

component will integrate different elements from the GUI of the other two components in order to define a unified wizard interface (Figure 18).

This graphical interface will allow the wizard to simulate a multimodal interaction based on complementarity of two simulated modalities. In order to generate combined events, the wizard has two choices. Firstly, s/he can push a button to put the WoZ component in a certain mode. For example, if s/he pushes the "zoom in" button, s/he will put the component in "zoom in" mode, and each time s/he will click on the map, s/he will generate a zoom command on that point. Secondly, s/he can use key shortcuts to simulate a multimodal interaction using the keyboard and the mouse. S/he clicks on the map, and generates a command using a key. On Figure 18, we can see that the wizard assigned the 'H' key to the "zoom here" command.

5. CONCLUSION

In this paper, we presented a summary of the different solutions for multimodal interaction that have been defined based on the CARE properties. The CARE properties (Complementarity, Assignment, Redundancy and Equivalence) define various forms that multimodal input interaction can take. Redundancy and Complementarity describe relationships between modalities and require fusion mechanisms. The CARE properties have been shown to be useful concepts for the design and evaluation of multimodal interaction in [8]. In this paper we showed how we make such concepts explicit while designing/implementing multimodal interaction and in particular for the cases of fusion (Redundancy and Complementarity). Two main approaches have been presented: a single fusion mechanism anchored in the PAC-Amodeus software architecture and a component-based approach. As part of our component-based approach for multimodal interaction, we introduced a new contribution based on generic composition components and WoZ composition components.

Future work will focus on other temporal aspects of the WoZ composition components, such as the integration of high-frequency data. We are currently studying the synchronous aspects of WoZ components for the case of simulating modalities that provide continuous events. Our goal is to avoid asking the wizard to perform repetitive tasks such as generating the same event at a fixed frequency. A further complementary challenge we plan to address is to define tools for analyzing the test data in the context of our component-based approach for rapidly prototyping and testing multimodal interaction.

6. REFERENCES

[1] Allen J., (1983). Maintaining Knowledge about Temporal Intervals. Journal Communication of the ACM 26(11), Novembre 1983, pp. 832-843.

[2] Bernsen, N. Modality Theory in support of multimodal interface design. Proceedings of Intelligent Multi-Media Multi-Modal Systems (1994), pp. 37-44.

[3] Bolt, R. A. (1980). Put-that-there: voice and gesture at the graphics interface. SIGGRAPH'80, 14, 3, pp. 262-270.

[4] Bouchet, J., Nigay, L. ICARE: A Component-Based Approach for the Design and Development of Multimodal Interfaces, *Extended Abstracts CHI'04* (2004), 1325-1328

[5] Bouchet, J., Nigay, L. (2004). ICARE Software Components for Rapidly Developing Multimodal Interfaces. In Proc. of ICMI'04, ACM Press, pp. 251-258.

[6] Cheng, H., et al. (2004). A Wizard of Oz Framework for Collecting Spoken Human-Computer Dialogs. Proc. Intl. Conf. on Spoken Language Processing,.

[7] Coutaz, J. PAC, an Object Oriented Model for Dialog Design, in Proc. Interact'87, pp. 431-436.

[8] Coutaz, J., Nigay, L. Salber, D., Blandford, A, May, J., Young, R.M. (1995). Four easy pieces for assessing the usability of multimodal interaction: the CARE properties. In Proc. of INTERACT'95, ACM Press, pp. 115-120.

[9] Davis, R., Saponas, T., Shilman, M., Landay, J. (2007). SketchWizard: Wizard of Oz Prototyping of Pen-Based User Interfaces. Proc of UIST'07, ACM Press, pp. 119-128.

[10] EJB. http://java.sun.com/products/ejb/.

[11] Interface-Z, www.interface-z.com.

[12] Landragin, F. (2007). Physical, Semantic and Pragmatic Levels for Multimodal Fusion and Fission. In Proc. of IWCS-7, pp. 346-350.

[13] Miki, M. et al. (2008). An integrative recognition method for speech and gestures. In Proc. of ICMI'08, ACM Press, pp. 93-96.

[14] Nigay, L., Coutaz, J. (1995). A Generic Platform for Adressing the Multimodal Challenge. Proc. of CHI'95, ACM Press, pp. 98 – 105.

[15] OMG - CORBA. www.corba.org

[16] OpenInterface European project. IST Framework 6 STREP funded by the European Commission (FP6-35182). www.oi-project.org.

[17] Oviatt, S., et al. (2003). Toward a Theory of Organized Multimodal Integration patterns during Human-Computer Interaction. In Proc. of ICMI'03, ACM Press, pp. 44-51.

[18] Oviatt, S., Lunsford, R. and Coulston, R. (2005). Individual Differences in Multimodal Integration Patterns: What Are They and Why Do They Exist?. In Proc. of CHI'05, ACM Press, pp. 241-249.

[19] Pfleger, N. (2004). Context Based Multimodal Fusion. In Proc. of ICMI'04, ACM Press, pp. 265-272.

[20] Serrano, M., Nigay, L. (2008). A Three-dimensional Characterization Space of Software Components for Rapidly Developing Multimodal Interfaces. Proc. of ICMI'08, ACM Press, pp. 149-156.

[21] Szyperski, C., Gruntz, D., Murer, S., "Component Software: Beyond Object-Oriented Programming, Second edition", ACM Press, ISBN 0-201-74572-0, 2002.

[22] The UIMS Tool Developers Workshop, A Metamodel for the Runtime Architecture of an Interactive System, SIGCHI Bulletin, 24, 1 (Jan. 1992), pp. 32-37.

[23] Vernier, F., Nigay, L. (2001). A Framework for the Combination and Characterization of Output Modalities. In Proc. of DSVIS'01, Springer Berlin / Heidelberg, Volume 1946/2001, pp. 35-50.