The OpenInterface Framework: A tool for multimodal interaction

Marcos Serrano
Grenoble Informatics Laboratory
University of Grenoble
Grenoble, France
marcos.serrano@imag.fr

Andrew Ramsay
Computing Science Department
University of Glasgow
Glasgow, UK
adr@dcs.gla.ac.uk

Laurence Nigay
Grenoble Informatics Laboratory
University of Grenoble
Grenoble, France
Laurence.nigay@imag.fr

Roderick Murray-Smith
Computing Science Department
University of Glasgow
Glasgow, UK
rod@dcs.gla.ac.uk

Jean-Yves L. Lawson
Université Catholique de Louvain
Louvain, Belgium
jean-yves.lawson@uclouvain.be

Sebastian Denef
Fraunhofer FIT
Sankt Augustin, Germany
sebastian.denef@fit.fraunhofer.de

CHI 2008, April 5 – April 10, 2008, Florence, Italy
ACM 978-1-60558-012-8/08/04

Abstract
The area of multimodal interaction has expanded rapidly. However, the implementation of multimodal systems still remains a difficult task. Addressing this problem, we describe the OpenInterface (OI) framework, a component-based tool for rapidly developing multimodal input interfaces. The OI underlying conceptual component model includes both generic and tailored components. In addition, to enable the rapid exploration of the multimodal design space for a given system, we need to capitalize on past experiences and include a large set of multimodal interaction techniques, their specifications and documentations. In this work-in-progress report, we present the current state of the OI framework and the two exploratory test-beds developed using the OpenInterface Interaction Development Environment.

Keywords
Multimodal Interaction, development environment, prototyping, software component.

ACM Classification Keywords
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

OpenInterface Project

The OpenInterface Project [11] is an ongoing European IST Framework 6 STREP (Specific Targeted Research Project) funded by the European Commission. The project is multidisciplinary and involves ten European academic and industrial partners including HCI researchers/practitioners, designers, psychologists, ergonomics and software engineers. The project started on September 2006 and ends on May 2009.

www.oi-project.org

Introduction

Even though multimodal interaction has expanded rapidly and real multimodal applications are being implemented, the design and development of multimodal interfaces remains a difficult task. This is especially true in mobile/ubiquitous environments or with novel interaction techniques. Available tools for developing interfaces are still designed for a limited set of techniques and interaction paradigms.

In this context, the aim of the OpenInterface project is to provide better ways of developing systems that use multimodal interaction. We are developing tools and methods for multimodal interaction development driven by exploratory test-beds for navigating large information spaces and multi-player mixed-reality games on mobile phones. In this paper, we introduce the OpenInterface framework, the generic and tailored components that have already been implemented and two test-beds we have built using our framework.

Related work

To date, most tools are designed for specific interaction paradigms, such as camera-based interaction [9] or tangible interfaces [7]. Concerning component-based toolkits for multimodal interaction, two main references have guided our study. The first is the Input Configurator [5] (ICON), which is a tool for building input reconfigurable interfaces. The main limitation of ICON is that the level of abstraction of its components is too low and assemblies of components become too complex. The second is our ICARE tool [2], which is our previous attempt to create a high level component-based approach for building multimodal interaction. ICARE defines a set of generic components. Our past experience showed that only a limited set of those components were really generic. In the OpenInterface framework, we therefore adopt a more realistic mixed approach that enables the designer/developer to use both generic components and tailored components. Tailored components are designed for specific application needs, while generic components implement high-level reusable functionalities.

OI framework structure

The OpenInterface (OI) framework is composed of the OI Kernel, a component-based runtime platform, and the OpenInterface Interaction Development Environment (OIDE), a graphical tool built on top of the platform.

OI Kernel

The OI Kernel [1] is a component-based platform that handles distributed heterogeneous components based on different technologies (Java, C++, Matlab, Python, .NET). The heterogeneity of the platform constitutes a great advance from previous approaches. It will allow the integration of existing interaction techniques written in different languages. The platform includes a tool for creating new components from existing code without modifying the original code. The OI kernel manages the creation and the connection between components by dynamically loading and interpreting the OI application descriptions or “pipelines”. Pipelines can be created graphically using the OIDE, built on top of the OI kernel.

OIDE

The OIDE is a graphical environment that allows direct manipulation and assembling of components in order to specify a "pipeline” defining a multimodal interaction. Figure 1 gives an example of a pipeline.

Dynamic assembly
The OI kernel and the OIDE allow the dynamic assembly of components: components can be launched individually. As seen in figure 1, each component has a button that allow a designer to start the component.

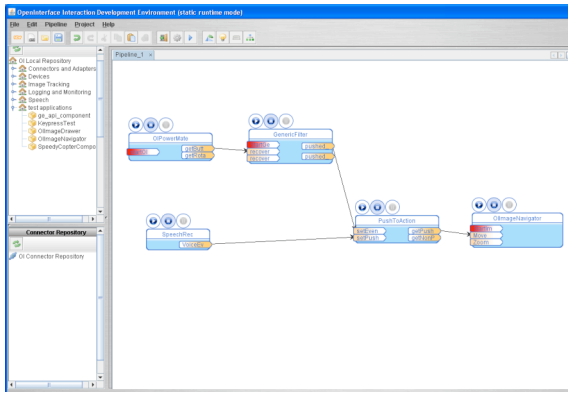


Figure 1. OpenInterface Interaction Development Environment (OIDE): A graphical editor for assembling components.

Component Repository
Already available components include device drivers, interaction techniques, multimodal fusion facilities, development services and developer-defined combinations. OIDE supports descriptions, querying and access in an extensible set of description schemas.

OI Conceptual Component-based model

Our model defines a set of generic and tailored components. Generic components define reusable generic operations. Tailored components are components implemented in an ad-hoc way for a specific interaction technique or interactive application.

This adopted mixed approach improves the expressive leverage of the OI framework.

Our model includes three main levels for components: devices, transformation chain and tasks, as shown in Figure 2. Both task and device components can be generic or tailored. For example generic task components are based on Foley's interactive tasks [6] while generic device components are based on device taxonomies [3]. The transformation chain level defines two types of components: transformation components and composition components.

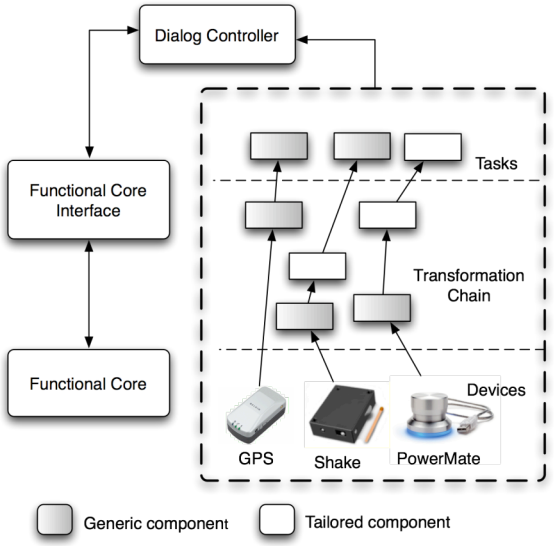
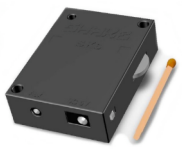
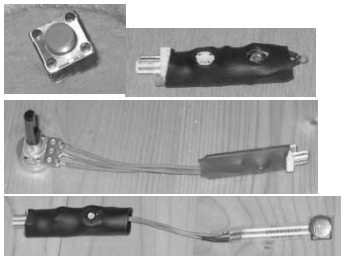


Figure 2. The OI model within an ARCH software architecture.



The Shake [12] packages a variety of sensors, including a triple axis accelerometer, two capacitive sensors and a vibrotactile actuator.

The Track IR is an IR camera-based tracker.



The Interface-Z [8] contains several innovative sensors, such as an atmospheric pressure sensor.

OI Components: Tools and Techniques

The OI framework has so far been populated with a variety of construction tools and interaction techniques in order to improve the development of multimodal interfaces.

Construction tools

The OI framework offers a set of debugging and monitoring tools to help designers in building multimodal applications. Those tools include an oscilloscope component that displays multiple continuous data streams (figure 4), a DataViewer component that can display numeric and textual data, a logging component that saves data streams in files, timestamping the data events, and a Generator component that can generate events based on mathematical formula.

Device and Tasks components

The OI framework has already been populated by a large set of interaction techniques at different levels of abstraction, such as speech recognition or image analysis (ARToolkit, TrackIR), and devices including the SHAKE [12], the interface-Z [8], the Wiimote, GPS units, the Diamond Touch [4], mice and keyboard. Figure 3 shows the representation of two devices. For each of them a tailored component is defined corresponding to the driver of the device. We then define three generic components that partially describe the various captured events.

Transformation components

The transformation chain of our conceptual model is composed of transformation and composition components. For each type of component, we aim at defining generic reusable components.

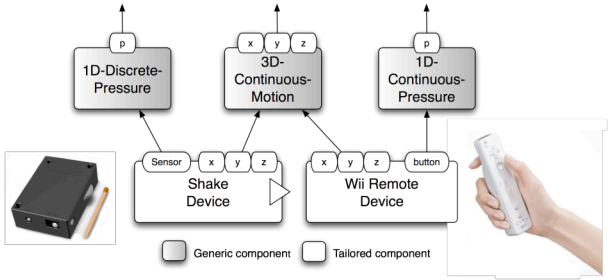
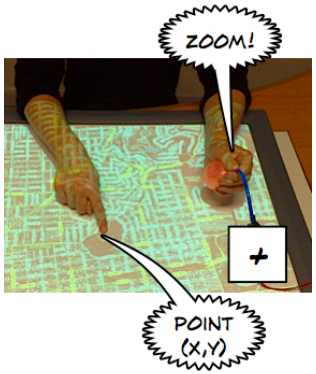


Figure 3. Device components: Shake and Wiimote devices.

Generic Composition components are based on our previous work ICARE. We define four types of transformation based on the CARE properties (Complementarity, Assignation, Redundancy and Equivalence) [10]. Those composition components perform temporal fusion of input modalities. Future work will focus on other types of composition, such as spatial or semantic composition.

Generic transformation components implement reusable operations that are usually performed on specific devices. For now we have identified three generic components: Command Filter, Threshold and Continuous-Discrete components. The Command Filter component generates a command from an input according to a mapping table. For example, we can generate commands from 3D gestures performed using the Wiimote. The Threshold component operates a filter on input values according to a configurable threshold. This component allows fine tuning of the interaction. The Continuous-Discrete component modifies the frequency of the input events. This modification can completely change the interaction: if we use the gesture and the shake for zooming, we would be able

Multimodal Map Navigation using the Diamond Touch and speech commands.



Multimodal Map Navigation using the Diamond Touch and the balloon of the interface-Z. The user can zoom by pressing the balloon.

to transform a continuous zoom (moving the hand down means continuous zoom-in) into a discrete zoom (moving the hand down means one zoom-in action).

Multimodal Map Navigation

Using the OI framework, we built a multimodal map navigation application. The user can navigate and zoom using several interaction techniques, such as (speech and gesture) or (pressure and gesture). Pressure is achieved using the interface-Z sensor.

The use of the OI framework for building the multimodal interaction of the map navigator offers many advantages to both designers and developers. For example, the developer can replace the Diamond Touch component by the Mouse component in order to perform tests on a simple computer. The designer can easily modify the interaction, as part of a user-centered design process, for example by tuning the intensity of the pressure of the balloon for zooming.

Gaming on a Mobile Phone

Using the OI framework, we built a 3D game on mobile phone. The main goal of the "Funny Rabbit" game is to find a rabbit that is hidden in a chest. The game runs on a mobile phone (Figure 4), while the different interaction techniques are hosted by a PC running the OI kernel. We use a variety of interaction techniques to control the game: speech commands, 3D gesture commands with the Shake and 3D gesture commands with the camera and ARToolkit.

The interaction can be rapidly modified using the OIDE. For example, the Shake component can easily be replaced by the ARToolkit component, implying minor modifications of the assembly of components. This improves the iterative player-centered design of the multimodal interaction of the game.

Figure 4. 3D Game on mobile phone with the Shake and oscilloscope visualization tool.

Conclusion and Future Work

The long term goal of this work is to define a flexible and integrating software framework for providing better ways of developing multimodal interaction based on generic and tailored components. We have already developed two test-beds, one on a game and one on a map-based application which consider two settings, namely an augmented environment and a mobile platform. Ergonomic evaluation has been performed and we further plan participatory design activities using our framework.

Concerning the framework, our future plans include creating a repository of high level interaction technique descriptions, actual instances of these techniques (OI

Acknowledgements

The authors thank the contribution of their OpenInterface colleagues:

- Arcadia Design (It)
- France Telecom (Fr)
- Fraunhofer Institute (De)
- Immersion (Fr)
- Multitel (Be)
- Phonoclick (Tr)
- TXT e-solutions (It)
- Université Catholique de Louvain (Be)
- Université J. Fourier (Fr)
- University of Glasgow (Uk)

kernel pipelines) and component descriptions. This repository will be accessible through a web interface, but will also be embedded in the OIDE to allow designers and developers to search the repository and extract relevant components or techniques for use in their application. Similarly, it will be possible to import data into the repository from the OIDE, meaning that after a new multimodal application has been created, it can easily be published in the repository and made available to all OIDE users.

Finally we also continue to enrich the framework with new components. In our approach we aim at integrating existing toolkits. ARToolkit [1] has already been embedded in our software framework and we are currently working on the integration of the phidgets [7]. We welcome contributions from the CHI community as designers of innovative multimodal interaction, as users of the platform for developing a multimodal application, or as developers of new components: new devices, new interaction techniques or new specifications of generic components for multimodality. The open source platform and the tool to add new components are free to download at forge.openinterface.org.

References

[1] Benoit, A., Bonnaud, L., Caplier, A., Damousis, I., Tzovaras, D., Jourde, E., Nigay, L., Serrano, M. and Lawson, J-Y. 2006. Multimodal Signal Processing and Interaction for a Driving Simulation: Component-based Architecture. Journal on Multimodal User Interfaces,1, 1,49-58.

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

[3] Buxton, W. (1983). Lexical and Pragmatic Considerations of Input Structures. ACM SIGGRAPH CG, 17(1), pp. 31-37

[4] Dietz, P. and Leigh, D. (2001). DiamondTouch: a multi-user touch technology. Proc. of the 14th annual ACM symposium on User interfaces software and technologies, UIST'01, pp. 219-226.

[5] Dragicevic, P., and Fekete, J. D. (2001). Input device selection and interaction configuration with ICON. Joint Proc. of IHM'01 and HCI'01,Springer Verlag, pp. 543-558.

[6] Foley, J., Wallace, V.L. and Chan, P. (1984) . The human factors of graphics interaction techniques. *IEEE Computer Graphics and Applications*, 11, pp. 13-48

[7] Greenberg, S. and Fitchett, C. (2001). Phidgets: Easy Development of Physical Interfaces throught Physical Widgets. In Proc. of the UIST 2001 Annual ACM Symposium on User Interface Software and Technology, ACM Press

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

[9] Maynes-Aminzade, D., Winograd, T., and Igarashi, T. (2007). Eyepatch: Prototyping Camera-based interaction through Examples. In Proc. UIST'07

[10] Nigay, L., Coutaz, J. (1997). Multifeature Systems: The CARE Properties and Their Impact on Software Design. Intelligence and Multimodality in Multimedia Interfaces, AAAI Press.

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

[12] Williamson, J., Murray-Smith, R., and Hugues, S. (2007). Shoogle: Ecitatory Multimodal Interaction on Mobile Devices. In Proc. CHI'07, ACM Press, pp. 121-1