EMF: Extensible Middleware Framework

Tiago Guerreiro[1], Vitor Fernandes[1], Bruno Araújo[1], Joaquim Jorge[1], João Pereira[1]

[1] Intelligent Multimodal Interfaces, INESC-ID,
R. Alves Redol, 9, 1000-029, Lisbon, Portugal,
{tjvg, vmnf, brar, jaj, jap}@immi.inesc.pt

Abstract. We present a framework to manage multimodal interfaces and applications in a reusable and extensible manner. We achieve this by focusing the architecture both on devices' Capabilities and users' Intentions. One particular domain we want to approach is collaborative environments where several modalities and applications make it necessary to provide for an extensible system combining diverse components across heterogeneous platforms on-the-fly.We demonstrate how to connect different applications and users around an immersive environment (tiled display wall), including different non-conventional input modalities and applications.

Keywords: Framework, Multimodal Interfaces, Extensibility, Reusable, Capabilities, Intents, Collaborative.

1 Introduction

Traditional interface devices in the HCI (Human Computer Interaction) area, basically the mouse and the keyboard, are still overwhelming before the emergent ways of interaction. However, in the last decades great efforts were made with promising results to present interaction options between persons and any type of computer. Besides, several projects have been presented to fuse input modalities and build natural multimodal applications [2]. Despite the availability of several non-conventional modalities very few applications take advantage of these technologies due to its implementation cost. In an environment where several multimodal applications are used, the developers' effort is often wasted due to its rigid focus on a certain application. Our framework provides the capacity to manage input modalities and capabilities accordingly with the user's will. We focus on reusing input capabilities and being able to add new modalities and applications with a few amount of effort. On the other hand, focusing our architecture both on devices' Capabilities and users' Intentions, we enable its use in collaborative environments, where any input is managed accordingly with the application and the capabilities available at the given moment. Finally, to demonstrate our framework, in a collaborative context, we present an immersive environment with a large display wall and several input modalities.

2 Framework

The main goal of our framework is to manage input modalities and applications separately allowing that each component can be reused and extended. We focus on inputs' capabilities and users' interests offering at each moment the most suited input for a determined task in a specific application. For a given input, the capabilities are the set of tokens or input data that can be offered by the input to any application, i.e., mouse commands, gesture tokens, tracking positioning or speech commands. On the other hand, a given token can be delivered by several input types, for example an "up direction" command could be given by a keyboard input or tracked body gesture motion. The framework is responsible to redirect needed tokens from the inputs to the application according to its preferences. Following this, different modalities can be integrated and shared between applications. Moreover, a new input which is able to deliver a known token for an application can be integrated easily extending the modalities without any change on the application.

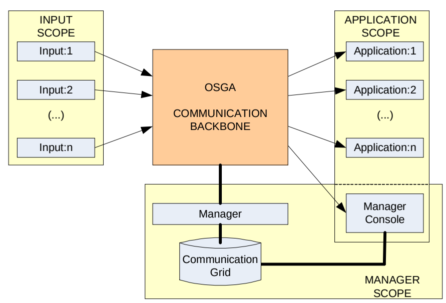

Fig. 1. - System Architecture

2.1 System Overview

In order to accomplish our extensibility goals, we used a message-oriented approach. Our system architecture, such as depicted by Fig. 1., is organized into four different entities: Inputs, Applications, a Manager and a communication backbone called OSGA. The Inputs are the communication interfaces for devices such as keyboard, mouse, speech recognition system, EMG (electromyographic signal capture), camera tracking systems which are able to deliver multimodal tokens corresponding to gestures commands or even data files. The Applications are linked to the communication backbone through an interface to allow the interaction using the input information. The communication backbone is responsible to redirect all the messages, allowing multiple Inputs to interact with multiple Applications. Doing so, Inputs can publish their data that will be received by all applications which have subscribed to it. The communication and the inputs' discovery process is coordinated by the fourth component: the Manager. This module supports the configuration and establishment of communications between Inputs and Applications. The coordination relies on a connection grid mechanism which stores the capabilities available for each connected input and the needs or preferences of each application. Finally, the Manager offers a user interface which allows the user switching and choosing inputs for applications. This interface takes advantage of existing inputs such as the applications do.

2.2 Open Source Group Architecture

Having an overwhelming number of prototypes emerging from investigation work, enables one to devise future integration projects bringing together several pieces to create larger and integrated applications. **OSGA (Open Source Groupware Architecture)** is a distributed XML message-based integration framework developed within our research group to overcome the integration problem. This framework can be easily used for further applications and can be integrated in our current prototypes. It is built upon XmlBlaster and provides the capacity to have several clients receiving messages accordingly to both subscription and publishing mechanisms. Basically, all the messages sent to the system are redirected to all the clients which subscribed to a given message topic. It also enables to filter the messages for a topic taking in account additional message properties such as sender’s identification, or clues about the message content.

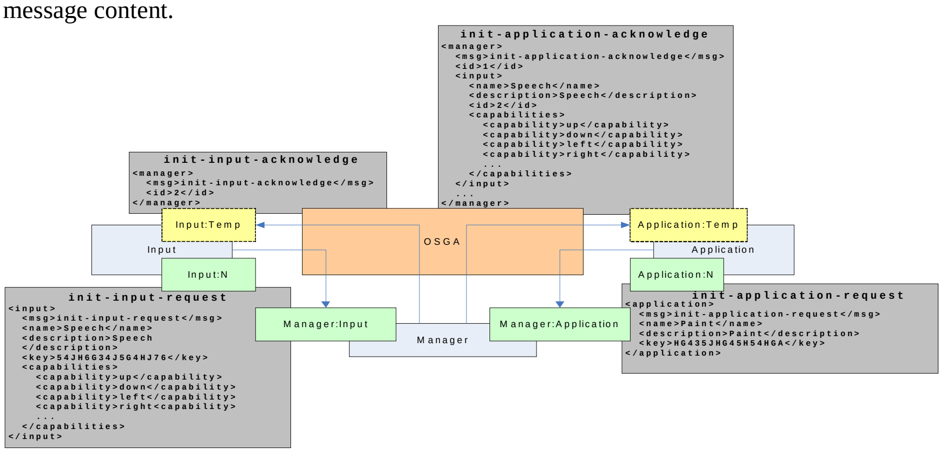

Fig. 2. – Communication Protocol

2.3 Communication Protocol

To support the interaction between Inputs, Applications and the Manager, we organized our XML messaging protocol (Fig. 2.) into four separated phases. The first step is the initialization of Input to register new inputs and its capabilities in the Manager. There is also the Application initialization to register the capabilities needed by the application and to know which inputs support these capabilities. The third phase is the communication setup to establish the communication between a given input and an application in order to receive input data. Finally, the protocol also supports the suspension/resume of a communication between an input and an application and its termination which is coordinated by the Manager.

Initialization: The Input initialization is performed by an init-input-request message using a temporary communication channel. Each input provides its identification and the list of supported capabilities. This information is stored by the Manager which will be available for interested applications. Regarding applications they also require an initialization in order to be identified by the Manager. On the other hand, the list of needed capabilities is provided and stored in the communication grid. The Manager provides a list of the most suited inputs to satisfy the capability needs by the application. This information will be used by the application to establish the connection with needed inputs to support the multimodal interaction.

Communication and Interaction: Applications can require, at any given time, the refereed list of inputs and capabilities, and the Manager will retrieve the actualized version of it. With the gathered information, applications can select the desired capabilities accordingly to the users' preferences. Manager replies to Inputs informing that Applications are interested on receiving their data. Each Application only receives what it revealed interest on through the use of filters. The *communication link* is established between the Inputs and the Application that subscribed these Inputs' topics. With all the required topics subscribed we achieve a n:n communication between Inputs and Applications improved by restriction mechanisms (filters) and enriched descriptions (messages traded) that can be updated on the fly. Inputs publishes data to the given topic '{input type}:{id}:data'.

Suspend and Resume: Communication between an Input and an Application can be suspended by Application initiative through the cancellation of Input subscription. To resume communication it will be enough to subscribe the Input topic.

Terminate: Applications and Inputs can terminate their session at any time publishing that intention to the Manager. Manager publishes affected Inputs and Applications that they must, eventually, reduce published data or modify the received capabilities accordingly.

Keep-alive: To prevent failure situations when Inputs/Applications aren't able to communicate with the Manager we developed a keep-alive signaling between this entities and the Manager.

3. Application Context

The framework presented in this paper is to be the background for all interaction and application prototypes created in our immersive environment, LEMe Wall. Hence, besides providing a collaborative framework, we are also able to reuse all input prototypes created and adjust them on-the-fly to each application accordingly to our needs. LEMe Wall [1] is an intelligent distributed environment with a multiprojection Display Wall (4x3 tiled display) as the main component. The environment is complemented by a set of sensors and actuators that increase the interaction immersion and naturalness.

3.1 Interaction Modalities

Besides the immersion provided by the large display screen, LEMe Wall aims at offering the user a set of interaction capabilities provided by several mechanism and modalities. As the users can walk freely around the room interacting with the environment we equipped it with several input mechanisms like a sensor network with pressure sensors and ultra-sound sensors for positioning, a network of five cameras for body gesture tracking and microphones for voice interaction. Through these interaction devices we can have several individuals interacting simultaneously with the environment within the same and different scopes (applications). The use of the framework in our immersive environment makes possible for all input prototypes developed to be available to everyone. This is quite useful when we consider project integration and reusing others researchers effort. Actually, when any student / researcher designs his project, he can visualize all the available inputs and capabilities through the Manager Console. With a small amount of effort every new input / capability is added to the available ones and can be used by everyone.

4. Conclusions

We presented a framework to manage multimodal interfaces in a distributed environment focusing on the extensibility and reusability of input modalities. We achieve this goal by separating inputs and applications and managing their communication through a protocol over a message-based system. The main contribution of the presented idea is the possibility to adapt inputs and applications on-the-fly accordingly to the available capabilities and user's will at any given time. Collaborative scenarios are well suited with this framework focus and goals. We presented an application scenario in an immersive environment where the collaborative and multimodal advantages are huge.

Acknowledgements. This work was supported in part by European Commission grant IST-2003-004785 (IMPROVE).

References

1. Araújo, B., Guerreiro, T., Jorge, J., Pereira, J., Jota, R. Leme Wall: Desenvolvendo um sistema de Multi-projecção. Proceeding of 13º EPCG , Vila Real, Portugal, Oct 2005.
2. Flippo, F., Krebs, A., and Marsic, I. 2003. A framework for rapid development of multimodal interfaces. In Proc. of the 5t h ICMI 03. ACM Press, New York, NY, 109-116