SOFTWARE DESIGN AND DEVELOPMENT OF MUTIMODAL INTERACTION

Marie-Luce Bourguet
Queen Mary University of London

Abstract: The multimodal dimension of a user interface raises numerous problems that are not present in more traditional interfaces. In this paper, we briefly review the current approaches in software design and modality integration techniques for multimodal interaction. We then propose a simple framework for describing multimodal interaction designs and for combining sets of user inputs of different modalities. We show that the proposed framework can help designers in reasoning about synchronization patterns problems and testing interaction robustness.

Key words: multimodal software architectures; integration techniques; finite state machines; synchronization patterns; recognition errors.

1. INTRODUCTION

Recent developments in recognition-based interaction technologies (e.g. speech and gesture recognition) have opened a myriad of new possibilities for the design and implementation of multimodal interfaces. However, designing and implementing systems that take advantage of these new interaction techniques is difficult. On one hand, our lack of understanding of how different modes of interaction can be best combined in the user interface often leads to interface designs with poor usability. On the other hand, developers still face major technical challenges for the implementation of multimodality, as indeed, the multimodal dimension of a user interface raises numerous challenges that are not present in more traditional interfaces. These new challenges include: the need to process inputs from different and heterogeneous streams; the co-ordination and integration of several

communication channels that operate in parallel *(modality fusion);* the partition of information sets for the generation of efficient multimodal presentations *(modality fission);* dealing with uncertainty and recognition errors; and implementing distributed interfaces over networks (e.g. when speech and gesture recognition are performed on different processors).

One of the main multimodal challenges lays in the implementation of adapted software architectures that enable modality fusion mechanisms. In this paper, we briefly review the current approaches in software design and modality integration techniques (section 2). Then in section 3, we propose a simple framework for describing multimodal interaction designs and for combining sets of user inputs of different modalities. In particular, we show that the proposed framework can help designers in reasoning about synchronization patterns problems and testing interaction robustness.

2. SOFTWARE ARCHITECTURES AND MODALITY FUSION TECHNIQUES

When implementing a multimodal system, several design and architectural decisions have to be made. The interdependency of input modalities and therefore the need for their integration in the system architecture can take several forms and fulfill different roles: redundancy (e.g. speech and lip movements), complementarity (e.g. "delete this" with a pointing gesture), disambiguation, support (e.g. speech and iconic hand gestures), modulation (e.g. speech and facial expressions), etc. Depending on the type of interdependency, a developer must try answering the following questions:

- At which level of granularity should data be processed on each input stream?
- How should heterogeneous information be represented?
- According to what criteria should modality integration be attempted?

Modality integration is usually attempted at either the feature (low) or the semantics (high) level, in two fundamentally different types of software architectures. Feature level architectures are generally considered appropriate for tightly related and synchronized modalities, such as speech and lip movements (Duchnowski et al, 1994). In this type of architecture, connectionist models can be used for processing single modalities because of their good performance as pattern classifiers, and because they can easily integrate heterogeneous features (Waibel et al, 1994). However, a truly multimodal connectionist approach is dependent on the availability of multimodal training data and such data is not currently available.

When the interdependency between modalities implies complementarity or disambiguation (e.g. speech and gesture inputs), information is typically integrated at the syntactic or semantic levels (Nigay et al, 1995). In this type of architecture, current approaches for modality integration include frame-based methods, multimodal grammars and agent-based frameworks. In frame-based methods, data structures called frames (Minsky, 1975) are used to represent meaning and knowledge and to merge information that results from different modality streams (Johnston, 1998). The use of grammars to parse multimodal inputs takes its inspiration from previous work in speech and natural language understanding (Shimazu et al, 1995). Grammars are sets of rules that describe all possible inputs. The main advantage of the grammatical approach is its generality, as grammars can be declared outside the core of a mutimodal system. Its main drawback lies in the difficulty of declaring the grammar without imposing constraints on users' behavior since a grammar must encompass all legal multimodal messages that a system will find acceptable. Finally, The agent based framework approach employs multiple agents to co-ordinate distributed information sources. In Martin et al (1999) for example, the framework is based on the Open Agent Architecture (OOA), a complex and general-purpose infrastructure for constructing systems composed of multiple software components. Agent based architectures are flexible and able to exploit parallelism.

3. PROTOTYPING MULTIMODAL INTERACTION

The models of architecture and integration techniques that can be found in the literature today and that were briefly reviewed in the previous section are often too generic or complex to provide ready-made solutions for developers. To date, no toolkit is available that addresses both the design and technical challenges of multimodality. In this section, we present a simple framework to support the designers and developers of multimodal user interfaces.

3.1 Designing Multimodality

Finite State Machines (FSMs) are a well-known technique for describing and controlling dialogs in graphical user interfaces (Wasserman, 1985). We show here that FSMs are also useful for modelling multimodal interaction and constitute a good framework for combining sets of user inputs of different modalities. Figure 1 illustrates how a speech and pen "move" command can be represented by an FSM.

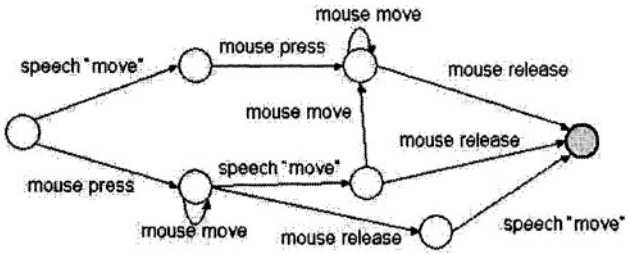

Figure 1. FSM modelling a "move" speech and pen command where many synchronisation patterns are represented.

When designing multimodal commands, one important task is the specification of the synchronization requirements. The aim is to guarantee that users will be able to activate the commands in a natural and spontaneous manner (Oviatt et al, 1997). In practice, a user can produce inputs in a sequential (e.g. with pen input completed before speech begins) or simultaneous manner (when both inputs show some temporal overlap). FSMs constitute a good framework for testing different synchronization patterns (Bourguet, 2003a). For example, Figure 1 describes a speech and pen "move" command where many different synchronisation patterns are represented. According to this representation, users are free to deliver inputs in their preferred order (sequentially or simultaneously, pen first or speech first). However, if we kept only the top branch of the FSM, users would become forced to use speech first and then the pen. Such an FSM would have for effect to constrain users in their usage of the modalities.

3.2 Testing interaction designs

Recognition-based technologies are still error-prone. Speech recognition systems, for example, are sensitive to vocabulary size, quality of audio signal and variability of voice parameters. In Oviatt (2000) it is shown that, during the process of semantic fusion, multimodal architectures can achieve automatic recovery from recognition errors and false interpretations. The phenomenon in which an input signal in one modality allows recovery from recognition error or ambiguity in a second signal in a different modality is called *mutual disambiguation* of input modes (Oviatt, 2000). However, the degree to which mutual disambiguation can operate in a given application is dependent on the design of the interaction, i.e. on the set of multimodal constructions that the system is able to interpret. In this section, we show that FSMs can help assessing the potential of different multimodal designs for mutual disambiguation of input signals (Bourguet, 2003b).

An FSM can naturally filter out erroneous recognition hypotheses because mis-recognised inputs that do not match the transition events of the current states can be ignored ("passive error handling"). The user may then choose to either repeat the input or reformulate it in order to increase the chances of good recognition. Once the input is correctly recognized, the dialog can resume. Passive error handling is a very simple technique that does not achieve error correction but is able to filter out erroneous recognition results. It is appropriate for testing the robustness of simple interaction models, where all FSMs are significantly different from each other.

When the recognition engine delivers more than one recognition hypothesis, an alternative strategy become possible. The event handler may dispatch subsequent recognition hypotheses until one is accepted by an FSM. In this case, the user does not need to repeat the input, as the recognition error has been automatically corrected. In order to work, this technique relies on the fact that the correct speech input is present in one of the recognition hypotheses.

The use of probabilistic state machines for dialog management for inputs with uncertainty has been discussed in Hudson et al (1992). This technique is relevant and applicable to multimodal interaction. The main difference between a probabilistic model and the traditional model of FSMs is that instead of having a single current state, a probabilistic FSM can have a distribution of alternative states. The probability that the machine is in any of these states is calculated based on a probability associated with each of the alternative user inputs. One potential advantage to this technique is that the probability of a state that triggered an action can be communicated to the application. The application can then combine this probability with its internal models to evaluate if an action should be executed or not, or to compare several concurrent actions.

4. CONCLUSION

The iterative design, implementation and testing of multimodal user interfaces is difficult, due to a lack of supporting tools for designers and developers. In response to this, we have developed a toolkit that aims to facilitate this process (Bourguet, 2002). In particular, modality integration, error handling and user input management are handled by the toolkit in a transparent manner. We have also developed a graphical tool that facilitates the process of declaring interaction models in the form of collections of FSMs (Bourguet, 2002). In the near future we are planning to automatically generate interaction models from experimental observations. Potential users

will be asked to freely produce actions with the aim of activating specific multimodal commands. These actions will then form the basis for the automatic generation of FSMs. These automatically generated FSMs will then be tested for error robustness using the techniques that were outlined in this paper.

REFERENCES

Bourguet, M.L., 2002, A Toolkit for Creating and Testing Multimodal Interface Designs, *in companion proceedings of UIST'02,* pp. 29-30.

Bourguet, M.L., 2003a, Designing and Prototyping Multimodal Commands, *in proceedings of INTERACT'03,* pp. 717-720.

Bourguet, M.L., 2003b, How finite state machines can be used to build error free multimodal interaction systems, *in proceedings of HCI'03 volume 2,* pp. 81-84.

Duchnowski, P., Meier, U. & Waibel, A., 1994, See Me, Hear Me: Integrating Automatic Speech Recognition and Lipreading, *in proceeding of ICSLP 94.*

Hudson, S, & Newell, G., 1992, Probabilistic Sate Machines: Dialog Management for Inputs with Uncertainty, *in Proceedings of UIST'92,* pp. 199-208.

Johnston, M., 1998, Unification-based Multimodal Parsing, *in proceeding of COLING-ACL.*

Martin, D., Cheyer, A. & Moran, D., 1999, The Open Agent Architecture: A framework for building distributed software systems, Applied Artificial Intelligence, **13** (1-2), pp 91-128.

Minsky, M., 1975, A framework for presenting knowledge, *The Psychology of Computer Vision,* P.H. Winston (ed.), McGraw-Hill.

Nigay, L & Coutaz, J., 1995, A Generic Platform for Addressing the Multimodal Challenge, *in Proceedings of CHI'95,* ACM Press, pp. 98-105.

Oviatt, S., De Angeli, A. & Kuhn, K., 1997, Integration and synchronisation of input modes during multimodal human-computer interaction, *in Proceedings of CHI'97,* ACM Press, pp. 415-422.

Oviatt, S., 2000, Taming recognition errors with a multimodal interface, *in Communications of the ACM,* **43** (9), ACM Press, pp. 45-51.

Shimazu, H. & Takashima, Y., 1995, Multimodal Definite Clause Grammar, *Systems and Computers in Japan,* **26** (3).

Waibel, A. & Duchnowski, P., 1994, Connectionist models in multimodal human-computer interaction, *in proceedings of GOMAC 94.*

Wasserman, A., 1985, Extending State Transition Diagrams for the Specification of Human-Computer Interaction, *IEEE Transactions on Software Engineering,* **11** (8), pp. 699-713.