A Formal model to handle the adaptability of Multimodal User Interfaces

Nadjet KAMEL
Université de Moncton
Campus de Shippagan
N.-B. Canada
nadjet@umcs.ca

Yamine AIT AMEUR
LISI/ENSMA, BP 40109
86961 Fururoscope Cedex,
France
yamine@ensma.fr

Sid-Ahmed SELOUANI
Université de Moncton
Campus de Shippagan
N.-B. Canada
selouani@umcs.ca

Habib HAMAM
Université de Moncton and
Canadian University of Dubai
Campus de Moncton
N.-B. Canada
hamamh@umoncton.ca

ABSTRACT

In this paper we propose an approach for checking adaptability property of multimodal User Interfaces (UIs) for systems used in dynamic environments like mobile phones and PDAs. The approach is based on a formal description of both the multimodal interaction and the property. The SMV model-checking formal technique is used for the verification process of the property. The approach is defined in two steps. First, the system is described using a formal model, and the property is specified using CTL (Computation Tree Logic) temporal logic. Then, we assume that an environment changes such that at most one modality of the system is disabled. For this propose, *Disable* is defined as a formal operator that disables a modality in the system. The property is checked by using the SMV (Symbolic Model Verifier) model-checker on all systems resulting from desabling a modality of the system. The approach reduces the complexity of the model-checking process and allows the verification at earlier stages of the development life cycle. We apply this approach on a mobile phone case study.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*
; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Evaluation/methodology*
; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Interaction techniques*

Ambi-sys '08, February 11-14, 2008, Quebec, Canada

General Terms

Verification

Keywords

multimodal interaction, formal model, adaptability

1. INTRODUCTION

Multimodal interaction is a characteristic of human communication where humans always speak and use gestures to point to objects. With the development of new interactive devices and technologies, UIs were enriched by human-like interaction to provide Multimodal UIs. At present, these kinds of UIs are developed for many systems using mobile devices like PDAs and mobile phones. People interact with these systems using one or more modalities that are defined as interaction techniques. The choice given to the user to use one or more modalities improves accessibility for diverse users and usage contexts [13], and enhances the performance stability, robustness, as well as the efficiency of communication. Multimodal interfaces increase the adaptability of the systems to the environment changes. For example, the user may interact with the system by using *speech* or, if the environment is too noisy, the person use *press keys*.

Since the characterization of these multimodal UIs makes their development more complex, the development of new models and approaches to design and verify their properties becomes necessary.

Empirical studies have explored multimodal UIs. Some of them aimed at defining usability properties [5]. Others investigated multimodality for web applications [7] and for mobile computing devices [18] to evaluate the usability of their interfaces. These studies lack of formalism and the interfaces evalutaion is always done through experimentations. The aim of this paper is to study the verification of adaptability property of multimodal interfaces at the design step before implementing the system. We propose an approach for designing a multimodal UI satisfing the adaptability property and including the case where one modality

becomes disabled. This makes the system more adaptable to situations of interactive device failures. The objective is to answer, at the design step, the following question: If a given property is satisfied by the interface, is this property still satisfied by the interface when a modality becomes unavailable? The approach that we propose is based on formal models and can be used at the design step. We use transition systems to model the system corresponding to the user interactions. We use CTL temporal logic [4] to express the property, and the model-checking technique to verify whether the property is satisfied by the system or not. We propose a development life cycle that helps designers to obtain a system satisfying the required constraints. To illustrate our approach, we use a case study corresponding to an interface of a mobile phone.

This paper is organised as follows. Section 2 introduces the context of the study by defining multimodal modalities and their characteristics. Section 3 briefly presents the research work related to the verification of multimodal Interfaces. We introduce our approach in this section. Section 4 presents the details of our formal model for multimodal interaction. The *Disable* operator is defined in this section. Section 5 details our expression of adaptability property. Section 6 briefly presents the SMV model-checking technique that we use to validate our approach on the case study presented in section 7. Finally we give our conclusion and future work in section 8.

2. MULTIMODAL INTERFACES

Multimodal interfaces are characterized by several possibilities, defined as modalities, to interact with the user. This kind of interface is implemented in many systems. The freedom degrees offered to users to interact with the system enhances the system flexibility and adaptability to different kinds of users, such as people with special needs, as well as to dynamic environments.

The use of multiple modalities occurs at the cost of the development complexity of this kind of system. In [12], the authors give an overview of novel concepts, defined by researchers of the HCI community, for designing and developing multimodal systems.

As in traditional interfaces, several properties have been identified for multimodal interactive systems. Two classes of properties can be distinguished. The first class is related to the robustness of the software such as deadlock freedom, liveness. The second class is related to the usability of the system such as reachability, observability, and preemptiveness. In [5], the authors have identified a set of properties required to assert that a multimodal interactive system is usable: they have defined the CARE (Complementarity, Assignation, Redundancy and Equivalence) properties.

The user profile and context of use variations make changes in the environment of the interaction. This forces the UI developer to take into account new properties such as *adaptability*. The adaptability property states that the UI remains usable even if some change has occurred within the interaction environment, like the disability of an interaction device. This property is very useful for mobile systems.

3. VERIFICATION OF MULTIMODAL UIS

Most of the research work in the area of design and evaluation of human-computer interfaces is based on experimentation and tests [6] [17] [15]. The associated methods have high costs. Indeed, when the required property is not satisfied by the system, the designer must modify the system at all the stages of the software life cycle. This is due to the fact that evaluation is made after the implementation of the system. Our approach allows checking properties at the design level (early). We focus on the adaptability property of the interface to changes in the interaction environment. We define the adaptability according to the reachability property. The *reachability* property states that a given state in the system is reachable. We consider that when the reachability property is satisfied even if a modality is disabled, then the *adaptability* property is satisfied.

Our approach uses formal methods based on mathematical models. Few studies have applyied formal methods for developing multimodal interactive systems. These studies aim at demonstrating the suitability of the formal methods to model and verify multimodal properties. Petri nets were used to model the fusion of modalities in [14]. In [3], the authors use finite state machines to test different synchronization patterns of modalities. Formal tests using Lustre-Lutess were used in [8] and transition systems and temporal logic were used in [9]. In [1], the authors have used the B method to verify the CARE properties.

Our approach consists of using a formal model to

1. describe the multimodal interaction;
2. express the reachability property;
3. check the reachability property; and
4. disable one modality of the system.

The previous points (1, 2, 3 and 4) are shown on Figure 1 which presents the different steps set up in our approach. The satisfiability of the adaptability property is deduced from the satisfiability of the reachability property after a cycle of verification. At each cycle, the reachability is checked on a system having one modality disabled.

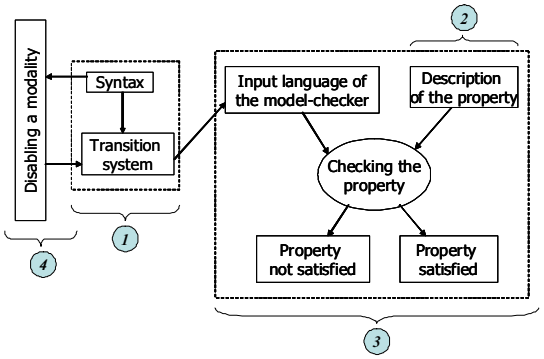

Figure 1: Our approach

The approach consists of two steps:

1. First, the designer describes the multimodal interactions using the formal model presented in Section 4; then he expresses the reachability property by CTL formula (see Section 5), and then he uses the SMV model-checker to verify whether the property is satisfied or not.

2. Second, the designer disables one modality using the *Disable* operator defined in Section 4.3, and checks the property again, on the new transition system, to verify whether it is still satisfied or not.

The designer can repeat the second step for all the modalities of the system or for any combination.

4. FORMAL MODEL FOR MULTIMODAL INTERACTION

Our formal model for the multimodal interaction system is based on the theory of interactive systems and process algebra developed by several authors [11]. Transition systems encoding the interactive system are used to formally represent this system. This formal description technique is universal and well understood. A transition system is used in a model-checking technique to represent the behavior of a system. It is basically a graph whose nodes represent the reachable states of the system and whose edges represent state transitions.

4.1 Syntax

The syntax of the language describing the multimodal interactions is given by the following grammar issued from classical process algebra. The rule defining S generates the multimodal user interface at a higher level by composing interaction events. The rule E generates the multimodal interaction event using basic interaction actions of the set A.

$$S ::= S[]S \mid S >> S \mid S \,|||\, S \mid S \,||\, S \mid E$$
$$E ::= a; E \mid a \,|||\, E \mid a \,||\, E \mid \delta \text{ with } a \in A$$

The symbols ; and >> stand for sequence operators. The symboles [], ||| and || stand, respectively, for choice, interleaving and parallel operators.

$(a1; a2; \delta[]a3; \delta)$ expresses the choice between two tasks. The first task is performed by the action $a1$ followed by the action $a2$. The second task is performed by the action $a3$.

4.2 Semantics

The underlying semantics of a multimodal system is a transition system. Let P and Q be two terms of the previous grammar and a, $a1$ and $a2$ be actions of A. Then, the transition $P \xrightarrow{a} Q$ expresses that the term Q is obtained from P when the action a is performed. $P \not\xrightarrow{a}$ expresses that the process P cannot perform the action a. In other words, there is no transition, labeled by a, starting from P. Using, this notation for transitions, the operational semantics is formally expressed by transition rules expressing the behavior of each operator of the previously described system. According to Plotkin [16], each rule of the form $\frac{premises}{conclusion}$ expresses that when the premises hold, then the conclusion holds. The formal semantics, given by the following set of rules, will be encoded according to the chosen formal used technique.

1. **Stop**. δ does not perform any transition;

2. **Prefix operator ;**. The term $a; P$ performs the action a and then behaves like the term P;

   $$a; P \xrightarrow{a} P$$

3. **Sequence operator** >>. If the term P performs an action a and then behaves like the term P' then, the term $P >> Q$ performs the same action and then behaves like the term $P' >> Q$. This is defined by the following rule :

   $$\frac{P \xrightarrow{a} P' \ and \ P' \neq \delta}{P >> Q \xrightarrow{a} P' >> Q}$$

   If the term P performs an action a and finishes then, the term $P >> Q$ performs the same action and then behaves like the term Q. This is defined by the following rule:

   $$\frac{P \xrightarrow{a} P' \ and \ P' = \delta}{P >> Q \xrightarrow{a} Q}$$

4. **Choice operator** []. The first rule asserts that if the term P performs an action a and then behaves like the term P' then, the term $P[]Q$ performs the same action and then behaves like the term P' (the second rule is defined in the reverse way).

   $$\frac{P \xrightarrow{a} P'}{P[]Q \xrightarrow{a} P'} \qquad \frac{Q \xrightarrow{a} Q'}{P[]Q \xrightarrow{a} Q'}$$

5. **Interleaving operator** |||. It allows interleaving (asynchronous) two transition systems (left and right). The first rule, asserts that if the left system transits from the state identified by the term P to the state identified by the term P', by performing an action a_1, then the composed state $P \,|||\, Q$ transits to $P' \,|||\, Q$ by performing the same action.

   $$\frac{P \xrightarrow{a1} P'}{P|||Q \xrightarrow{a_1} P'|||Q}$$

   In the same way, the second rule expresses, the behavior of the composed system resulting from the behavior of the right term (Q).

   $$\frac{Q \xrightarrow{a2} Q'}{P|||Q \xrightarrow{a_2} P|||Q'}$$

6. **Parallel operator** ||. It allows running two transition systems (left and right) in parallel (synchronous). The two first rules express the interleaving between actions.

$$\frac{P \xrightarrow{a_1} P'}{P||Q \xrightarrow{a_1} P'||Q} \qquad \frac{Q \xrightarrow{a_2} Q'}{P||Q \xrightarrow{a_2} P||Q'}$$

The third rule expresses that if the term P performs an action a_1 and behaves like the term P', and the term Q performs an action a_2 and behaves like the term Q' then $P \parallel Q$ performs the two actions in one step and behaves like $P' \parallel Q'$. We assume that the system, can not perform more than one action a_i, produced by the same modality, at the same time. For this, we use the function *mod* that assigns, to each interactive action, the modality that it produces it.

$$\frac{P \xrightarrow{a_1} P' \text{ and } Q \xrightarrow{a_2} Q' \text{ and } mod(a_1) \neq mod(a_2)}{P||Q \xrightarrow{(a_1,a_2)} P'||Q'}$$

Prefix, sequence, and choice operators, allow encoding sequential systems. When interleaving and parallelism operators are added, it becomes possible for different systems to behave in parallel. Moreover, these operations allow encoding synchrony and asynchrony. Indeed, ||| describes asynchrony, while || describes synchrony.

The transition system corresponding to any term of the grammar is obtained by applying the previous rules inductively.

4.3 Disabling operator: syntax and semantic

We define *Disable* as an operator that disables a modality in a system. As in the previous operators, the *Disable* operator is defined by syntax and semantics. Let P be a term of the grammar presented in the previous section, mi, one of the modalities of the system. $Disable(P, mi)$ is the term resulting from disabling the modality mi in the system P. The semantics of this operator is given by the following rules:

$$\frac{P \xrightarrow{a} P' \text{ and } mod(a)=mi}{Disable(P,mi) \not\xrightarrow{a}}$$

$$\frac{P \xrightarrow{a} P' \text{ and } mod(a) \neq mi}{Disable(P,mi) \xrightarrow{a} Disable(P',mi)}$$

The first rule states that if a term P performs an action a produced by the modality mi and behaves like P', then the term $Disable(P, mi)$ cannot perform the action a. It leads to a blocked state.

The second rule states that if a term P performs an action a produced by any modality other than the modality mi and behaves like P', then the term $Disable(P', mi)$ performs the same action and behaves like the term $Disable(P', mi)$.

5. THE ADAPTABILITY PROPERTY

We consider the property of adaptability which states that if the environment of the system changes, the interface remains usable. For instance, if the user can perform a given task using the *Speech* modality, and if the environment of the interaction changes, then the user must be able to use another modality to perform the same task. The property of adaptability can be defined by using the description of the *reachability* property, and expressed informally as: starting from a given state S_1, identified as the initial state of the task T, the user can reach a given state S_2, identified as the final state of the task T. In other words, it states that the user can perform the task T. This property can be expressed for a task T identified by its initial state S_1 and final state S_2, by the following generic CTL temporal logic formula:

$$\mathbf{AG}((state = initialState) \Rightarrow \mathbf{EF}(state = finalState))$$

state is a variable that may have a list of pairs $(variable, value)$. *initialState* and $finalState$ are state values corresponding, respectively, to the initial and the final state of the task T.

Informally this formula states that in all states (G) of all paths (A), if a state is identified as the initial state of the task $(state = initialState)$, then there exists (E) at least one path starting from this state and reaching, in the future (F), a state identified as the final state of the task $(state = finalState)$.

6. SMV MODEL CHECKER

The SMV model-checker [10] is a tool for checking finite state transition systems against specifications in the CTL temporal logic [4]. The transition system is described in the input language of SMV; the properties are specified in the temporal logic CTL that allows for the expression of a rich class of properties. The input file describes both the model and the specification. The model is a Kripke structure (transition system), whose states are defined by a collection of state variables, that may be of Boolean or scalar type. The transition relation of the Kripke structure, and its initial state, are determined by a collection of parallel assignments. They are introduced by the keyword ASSIGN. The specification of the system appears as CTL formulas under the keyword SPEC.

7. CASE STUDY

To validate our model, we model a mobile phone multimodal interface inspired from [2]. We assume that the user can interact with the interface by using *Speech* or by clicking on the press keys of the mobile phone to scroll and select menu items. A set of hierarchical menu items are specified allowing the user to access a set of tasks. A schematic view of a part of the interface is shown in Table 1.

7.1 Modelisation step

At this step, the designer determines the set of interactive actions and how they are composed through the use of the operators given in Section 4. The set A_{mi} defines all the interactive actions using the modality mi of the mobile phone system. We define two sets of interactive actions produced

Table 1: Some elements of the hierarchical menu

Menu
- Contacts
  - View All
  - Add New
    - Phone Number
    - Email Adress
    - ⋮
  - Recent Call
  - ⋮
- Settings
  - Silent
  - Sounds
  - ⋮

by the modalities $Speech$ and $PressKeys$.

$A_{speech} = \{"up","down","select"\}$
$A_{Keys} = \{C_{up}, C_{down}, OK, C_{right}, C_{left}\}$

where *"up"*, *"down"* and *"select"* are the words pronounced by the user and recognized by the system as actions that, respectively, move up, move down in the menu, and select an item. C_{Up}, C_{Down}, OK, C_{Right} and C_{Left} are actions of clicking, respectively, on the keys: Up, Down, Ok, right and left. These actions, respectively, allow the user to *move up*, *move down*, *select*, *open menu* and *close menu*.

7.1.1 Interaction model

Using the interactive actions defined in the previous section, the designer uses the formal model defined in Section 4 to describe the interaction model of the mobile phone. To illustrate our approach we focus on the following tasks $T1$, $T2$ and $T3$:

$T1 :$ "*View contact list*"
$T2 :$ "*select the ringer volume*"
$T3 :$ "*recent call*"

We suppose that initially, the cursor is on the $Menu$ item. The tasks $T1$, $T2$ and $T3$ consist, respectively, in viewing the contact list, selecting the ringer volume and showing the recent call.

The expression of each task using the interactive actions is given as follow:

$T1 : (C_{Down}; C_{right}; C_{Down}; OK)$
$\quad [] ("Contacts"; "viewlist"; "OK"')$
$T2 : (C_{Down}; C_{Down}; C_{Down}; C_{right}; C_{Down}; C_{Down}; OK)$
$\quad [] ("Settings"; "Sounds"; "RingerVolume"; "OK")$
$T3 : (C_{Down}; C_{Down}; OK)$

To perform the task $T1$ or $T2$, the user can use $Speech$ or $Keys$ on the press keys. The task $T3$ can be performed only by clicking on the press keys.

The expression of the interaction model is given as follows:

$MobilInteraction = T1[]T2[]T3$

It expresses that the user can perform the task $T1$ or $T2$ or $T3$.

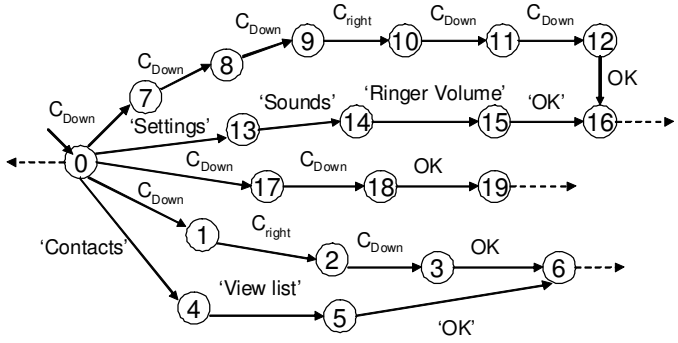

Figure 2: Transition system of MobilInteraction

By applying the semantic rules presented in Section 4.2, the transition system of each task ($T1$,$T2$,$T3$) is obtained. Then, by applying the semantic rule of the choice operator, the transition system of $MobileInteraction$ is obtained by composing those of the tasks $T1$, $T2$ and $T3$. The transition system of $MobilInteraction$ is given in Figure 2.

The state 0 is the initial state of both the tasks $T1$, $T2$, $T3$ and $MobileInteraction$. The states 1, 2, 3, 4, 5 and 6 are states of the task $T1$. The states 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 are states of the task $T2$. The states 17, 18 and 19 are states of the task $T3$.

The dashed arrows from the states 0, 6, 19 and 16 indicate that another task can be performed in alternative to, respectively, the tasks $T1$, $T2$ and $T3$; and another task can be performed after the tasks $T1$, $T2$ and $T3$ have finished.

7.1.2 Disabling modalities

We suppose that these two situations may occur:

1. the user is in a noisy environment and the modality $Speech$ is disabled. In this case, the user can only use the press keys to interact with the system. Then $Disable(ModelInteraction, Speech)$ describes this new system. The corresponding transition system is illustrated by Figure 3. It is obtained by applying the semantic rule of the operator $Disable$ presented in the Section 4.3, by replacing P and mi, respectively, by $ModelInteraction$ and $Speech$.

2. the user cannot click on press keys. In this case, he can only use $Speech$ to interact with the system. Then, $Disable(ModelInteraction, Keys)$ describes this new system. The corresponding transition system is illustrated by Figure 4. It is obtained by applying the semantic rule of the operator $Disable$ presented in the Section 4.3, by replacing P and mi, respectively, by $ModelInteraction$ and $Keys$.

In the two previous cases, we aim to check if the *reachability* property is always satisfied by the interface. For this propose, we use the SMV model-checker to verify if the property given in Section 5 is still satisfied. The verification process is given in the next section.

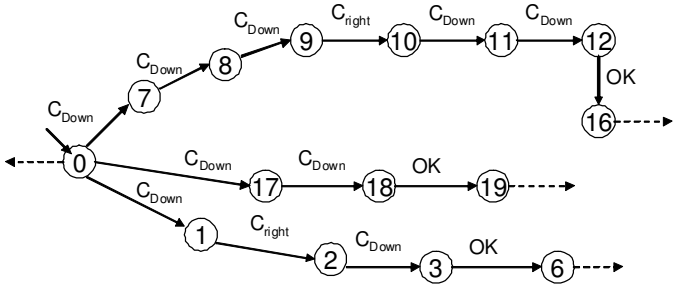

Figure 3: Disabling speech

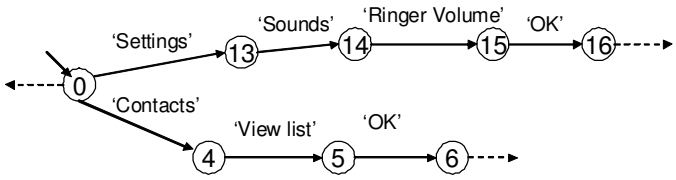

Figure 4: Disabling keys press

7.2 Verification step

Each of the transition systems of $ModelInteraction$, $Disable(ModelInteraction, speech)$ and $Disable(ModelInteraction, Keys)$ are translated to the input language of SMV.

In the clause VAR, two variables are defined to characterise the system states. The variable $State$ defines the global state of the system and the variable Mod defines the modality used to perform the interactive actions. $State$'s values are defined by the set 0..19 and Mod's values are defined by the set $\{Speech, Keys\}$.

In the clause ASSIGN, the SMV operator *next* is used to define the next value for both the variables $State$ and Mod, according to the transition system in encoding.

In the clause SPEC, the properties are expressed as CTL formulas. The property of the accessibility of the task $T3$ is expressed by the following formula:

$$\mathbf{AG}((state = 0) \Rightarrow \mathbf{EF}(state = 19))$$

This property is satisfied by the transition systems of both $ModelInteraction$ and $Disable(ModelInteraction, speech)$. Indeed, in the two cases, there exists a path starting from the state 0 and reaching the state 19. The transition system of $Disable(ModelInteraction, Clic)$ does not satisfy the property because there is no path starting from the state 0 and reaching the state 19. In this case, the designer deduces that the interaction model of the mobile phone as it is described in Section 7.1.1 is not adaptable to the changes of interaction environment.

8. CONCLUSION

This paper presented an approach to design multimodal interfaces that can adapt to changes in dynamic environments and to the failures of interactive devices. The approach is based on a formal model. The main advantage of this model is to allow the verification of multimodal interface properties at earlier stages of design. The approach is generic and can be used to verify any desired property that it must be independent from the availability of a given modality. The *Disable* operator enriches our formal model and its advantage is that it provides the transition system by disabling one modality without need to redesign the whole system. We note that the resulting transition system is less complex since the number of its states is reduced. This reduces the complexity of the model-checking process. We intend, in a future work, to use this operator to check CARE properties and functional equivalences between multimodal interfaces.

9. REFERENCES

[1] Y. Ait-Ameur, I. Ait-Sadoune, and M. Baron. Etude et comparaison de scénarios de développements formels d'interfaces multi-modales fondés sur la preuve et le raffinement. In *MOSIM 2006 - 6ème Conférence Francophone de Modélisation et Simulation. Modélisation, Optimisation et Simulation des Systèmes : Défis et Opportunités*, Rabat, 2006.

[2] R. Amant, T. Horton, and F. Ritter. Model-based evaluation of expert cell phone menu interaction. *ACM Transaction on Computer-Human Interaction*, 14(1):347–371, 2007.

[3] M. L. Bourguet. Designing and Prototyping Multimodal Commands. In *Proceedings of INTERACT'03*, pages 717–720, 2003.

[4] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 2(8):244–263, 1986.

[5] J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May, and R. Young. Four Easy Pieces for Asserting the Usability of Multimodal Interaction : the CARE properties. In *Proceedings of Human Computer Interaction - INTERACT'95*, pages 115–120, Lillehammer, Norway, June 1995. 'Bass, L. J. and Unger, C.

[6] H. B.-L. Duh, G. C. Tan, and V. H. hua Chen. Usability evaluation for mobile device: A comparison of laboratory and field tests. In *8th Conference on Human-Computer Interaction with Mobile Device and Services, Mobile HCI'06*, pages 181–186, Helsink, Finland, 2006. ACM 2006, ISBN 1-59593-5.

[7] M. Honkala and M. Pohja. Multimodal interaction with xforms. In *ICWE'06*, pages 201–208, Palo Alto, California, USA, 2006. ACM 2006, ISBN 1-59593-352.

[8] F. Jourde, L. Nigay, and I. Parissis. Test formel de systèmes interactifs multimodaux : couplage ICARE - Lutess. In *ICSSEA2006, 19ème journées Internationales "génie logiciel & Ingénierie de Systèmes et leurs Applications" Globalisation des services et des systèmes*, 2006.

[9] N. Kamel and Y. Ait-Ameur. Mise en oeuvre d'IHM Multimodales dans un système de CAO. Une approche fondée sur les méthodes formelles. *Revue internationale d'ingénierie numérique*, 1(2):235–256, 2005.

[10] K. M. Millan. The SMV System. Technical report, Carnegie Mellon University, 1992.

[11] R. Milner. A calculus of comunicating systems. volume 92, New York, NY, USA, 1980. LNCS, Springer-Verlag.

[12] L. Nigay and J. Coutaz. Espaces conceptuels pour l'intéraction multomédia et multimodale. *TSI, speciale multimedia et collecticiel*, 15(9):1195–1225, 1996.

[13] Z. Obrenovic, J. Abascal, and D. Starcevic. Universal accessibility as multimodal design issue. *Communication of The ACM*, 50(5):83–88, 2007.

[14] P. Palanque and A. Schyn. A model-based for engineering multimodal interactive systems. In *9th IFIP TC13 International Conference on Human Computer Interaction (Interact'2003)*, 2003.

[15] J. Pascoe, N. Ryan, and D. Morse. Using while moving; human-computer interaction issues in fieldwork environments. In *Annual Conference Meeting Transactions on Computer-Human Interaction*, pages 417–437, 2000.

[16] G. Plotkin. A Structural Approach to Operational Semantics. Technical report, Department of of computer Science, University of Arhus DAIMI FN 19, 1981.

[17] M. Serrano, L. Nigay, R. Demumieux, J. Descos, and P. Losquin. Multimodal interaction on mobile phones: Development an evaluation using acicare. In *8th Conference on Human-Computer Interaction with Mobile Device and Services, Mobile HCI'06*, pages 129–1136, Helsink, Finland, 2006. ACM 2006, ISBN 1-59593-5.

[18] R. Simon, F. Wegscheider, and K. Tolar. Tool-supported single authoting for device independence and multimodality. In *Mobile HCI'05*, pages 91–98, Salzburg, Austria, 2005. ACM 2006, ISBN 1-59593-089.