Human-Computer Interaction -- INTERACT'03
M. Rauterberg et al. (Eds.)
Published by IOS Press, (c) IFIP, 2003, pp. 543-550

A Model-Based Approach for Engineering Multimodal Interactive Systems

Philippe Palanque & Amélie Schyn

LIIHS-IRIT, Université Paul Sabatier (Toulouse III), 31062 Toulouse Cedex 4

palanque@irit.fr - schyn@irit.fr

Abstract: Representing the behaviour of multimodal interactive systems in a complete, concise and non-ambiguous way is still a challenge for formal description techniques. Indeed, multimodal interactive systems embed specific constraints that are either cumbersome or impossible to capture with classical formal description techniques. This is due to both the multiple facets of a multimodal system and the strong temporal constraints usually encountered in this kind of systems. This paper presents a formal description technique dedicated to the engineering of interactive multimodal systems. Its aim is to provide a precise way for describing, analyzing and reasoning about multi-modal interactive systems prior to their implementation. One of the basic components for multi-modal systems is the fusion mechanisms. This paper focuses on this component and, in order to exemplify the approach, the formal description technique is used for the modelling and the analysis of one fusion mechanism. Lastly, benefits and limitations of the approach are discussed.

Keywords: Formal description techniques, multi-modal interaction, Safety critical applications

1 Introduction

Despite some efforts for providing toolkits for the construction of multimodal interactive systems (Bederson et al., 2000; Chatty, 1994), the actual engineering of multimodal interactive systems remains a cumbersome task usually carried out in a rather crafty process. Indeed, while the design (Coutaz & Nigay, 1993; Nigay &Vernier, 2000) and the evaluation (Coutaz et al., 1996) of multimodal interactive systems have been thoroughly studied, the process of going from a given design to an actual functional system has been the focus of very little research work.

An important aspect of this development process is the reuse of work done from a previous design to another application. Some work on toolkits (Bederson et al., 2000) and architectures (Nigay & Coutaz, 1995) address this problem at a very low level of abstraction thus making the solution bounded either to modalities or to development platforms.

We believe that the use of an adequate formal description technique can provide support for a more systematic development of multimodal interactive systems. Indeed, formal description techniques allows for describing a system in a complete and non-ambiguous way thus allowing for an easier understanding of problems between the various persons taking part in the development process. Besides, formal description techniques allow designers to reason about the models by using analysis techniques. Classical results can be the detection of deadlock or presence or absence of terminating state. A set of properties for multimodal systems have been identified (Coutaz et al., 1995) but their verification over an existing multimodal system is usually impossible to achieve. For instance it is impossible to guarantee that two modalities remain redundant whatever state the system is in.

The aim of this paper is to present such a formal description technique. This proposal builds upon previous work we have done in the field of formal description techniques for interactive systems (see section 2) and is an answer to several requests from industry to provide software engineers with software engineering techniques for multimodal systems.

The paper is structured as follows. Next section is dedicated to related work dealing with specification of multimodal interactive systems. Section 3 is dedicated to the informal presentation of the ICO (Interactive Cooperative Objects) formalism. Section 4 presents extensions to ICO formalism which is dedicated to the formal description of multimodal interactive systems. This formalism is applied to one fusion mechanism, which integrates voice and gesture. Formal analysis of the fusion engine is also presented. Last section (section 5) presents the advantages and limitations of the approach as well as future and ongoing work.

2 Related work

Work in the field of multimodal can be sorted in five main categories. Of course, the aim of this categorization is not to be exhaustive but to propose an organization of previous work in this field.

- Understanding multimodal systems: (Coutaz & Nigay, 1993) presents a typology of multimodal systems while (Coutaz et al., 1995) deals with properties of multimodal systems.
- Software construction of multimodal systems: (Bederson et al., 2000) or (Chatty, 1994) propose toolkits for the construction of multimodal systems, and (Nigay & Coutaz, 1995) proposes a generic software architecture for multimodal systems.
- Analysis and use of novel modalities: (Bolt, 1980) presents the first use of voice and gesture as combined modalities. (Buxton & Myers, 1986) introduces two handed interaction (Bolt & Herranz, 1992) introduces the use of two handed interaction for virtual reality applications and (Vo & Wood, 1996) presents Jeanie, a multimodal application, to test the use of eye tracking and lips movements recognition.
- Multimodal systems description: (Cohen et al., 1997) presents QuickSet a cooperative interface using both voice and gesture. (Nigay & Coutaz, 1995) presents a Multimodal Air Traffic Information System (MATIS) using both voice and direct manipulation interaction. Similarly, (Bier et al., 1993) presents a drawing systems featuring two handed interaction through a trackball and a mouse.
- Multimodal systems modeling: (Accot et al., 1996) exploits high-level Petri nets for modeling two handed interaction (a mouse and a trackball) and (Hinckley et al., 1998) uses finite state automatons for modeling two handed interaction (a touchpad and a TouchMouse). While formal description techniques have been defined and used for interactive systems since the early work from Parnas (Parnas, 1969), their extension and use for multimodal systems is still relatively rare. We can quote for instance work from (Duke & Harrison, 1997) or (McColl & Carrington, 1998) were they present how software engineering techniques such as Z and CSP can be used for the modeling of MATIS the multimodal air traffic information system developed by Nigay (Nigay & Coutaz, 1995).

We believe that multimodal interactive systems feature intrinsic characteristics that make formal description techniques used in software engineering not directly suitable for multimodal systems. First, multimodal interactive systems are, by definition, interactive and thus behave in an event-driven way, usually hard to capture and represent in state-based descriptions such as Z. Second, the temporal constraints are at the core of these systems which are more often than not real time and highly concurrent. Indeed, users' actions may occur simultaneously on several input devices and the fusion mechanism must process those input in real-time. Formal description techniques with an interleaving semantics (such as CSP, CCS or LOTOS) are not capable of representing such truly concurrent behaviors. Lastly, the use of temporal windows in fusion mechanisms requires, from a formal description technique, the possibility to represent time in a quantitative way by expressing for instance that an event must be received within 0.1 second after the previous one.

Petri nets are one of the few formal description techniques that allows for representing the behavior of such systems (Bastide & Palanque 1995). Indeed, they feature true-concurrency semantics, they are able to deal both with events and states and they provide several ways to represent quantitative time (Bastide & Palanque, 1994). Besides they represent state in a concise way as states are represented in intention and not in extension (as in state machines for instance) thus avoiding combinatory explosion when combining several devices as this can be seen in (Hinckley et al., 1998).

3 Informal description of ICOs

The Interactive Cooperative Objects (ICOs) formalism is a formal description technique dedicated to the specification, modeling and implementation of interactive systems (Bastide et al., 1998). It uses concepts borrowed from the object-oriented approach (dynamic instantiation, classification, encapsulation, inheritance, client/server relationship) to describe the structural or static aspects of systems, and uses high-level Petri nets (Genrich, 1991) to describe their dynamic or behavioral aspects.

In the ICO formalism, an object is an entity featuring five components: a cooperative object (CO), an availability function, a presentation part, and two functions (the activation function and the rendering function) that make the link between the cooperative object and the presentation part.

The **Cooperative Object** (CO) models the behaviour of an ICO. It states (by means of a high-level Petri net) how the object reacts to external stimuli according to its inner state. The Petri model used in the ICO formalism is called a high-level Petri net model as token can hold values (such as references to other objects in the system). A Cooperative Object offers two kinds of services. The first ones concern the services offered to others objects of the system (called system services) and the second ones (called event services) concern the services offered to a user (producing events) or to other components in the system but only through event-based communication. The availability of all the services in a CO (which depends on the internal state of the objects) is fully stated by the high-level Petri net.

The **availability function** links a service to its corresponding transitions in the CO i.e. a service offered by an object will only be available if one of its related transition in the Petri net is available.

The **presentation part** presents the external appearance of the ICOs. It is a set of widgets in a set of windows and a set of graphical functions that can be called by the CO. Each widget can be used (by the user) for interacting with the ICO and/or a way (for the system) to display information about internal state of the object. The **activation function** links users' actions on the presentation part (for instance a click using a mouse on a button) to event services. The **rendering function** maintains the consistency between the internal state of the system and its external appearance by reflecting system states changes through functions calls.

An ICO model is fully executable, which gives the possibility to prototype and test an application before it is fully implemented (Bastide et al. 2002). The models can also be validated using analysis and proof tools developed within the Petri nets community and extended in order to take into account the specificities of the Petri net dialect used in the ICO formal description technique.

The description presented here integrates significant changes with respect to previous work done on the ICO formalism. These changes were required in order to be able to deal with multimodal interactive systems:

- We have added a event-based communication channel,
- We have added a structuring mechanism based on transducers in order to deal with low level and higher level events,
- We have added temporal modeling and interpretation in order to deal with temporal constraints,
- We have had to redefine the formal semantics of ICO in order to deal with those changes. This can be found here http://lihs.univ-tlse1.fr/palanque/ICOs.htm

4 Modelling Fusion Mechanisms

This section presents how the ICO formalism can be used for modelling fusion mechanisms in a multimodal interactive system.

4.1 Voice and Gesture Interaction

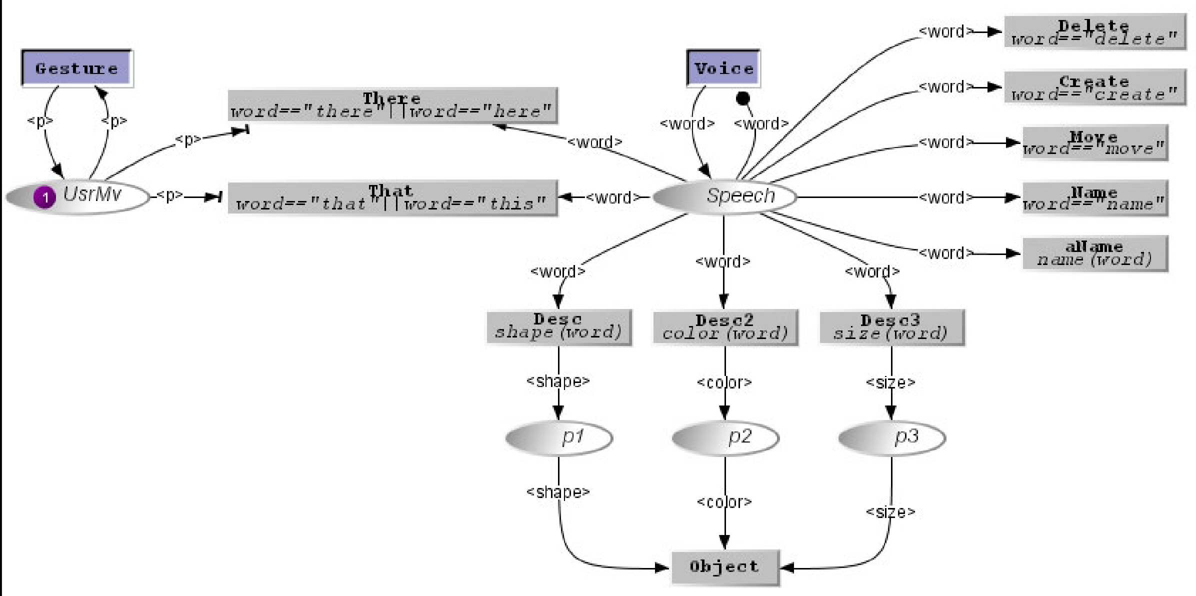
Our example is Bolt's system (Bolt, 1980). Bolt was the first to propose to use both voice and gesture recognition synergistically for multimodal input. This idea has been implemented in a drawing application, in which a user can enter a command orally and give the arguments using either a precise oral description or using a deictic word (this, that, there ...) together with a designation gesture.

In this system, five commands are available: Create, Name (to associate a name to an object), Delete, Move and Make (in order to save place Make is not represented in this paper). Each command features a given number of arguments each of them being of a given type. There is a strong dialog constraint in the system as it is described in Bolt's papers: as long as the command is incomplete, the system waits for the missing argument(s). This means that the system can be completely stuck if only one argument is missing.

Our activity on this case study has been to reengineer Bolt's system. As an input for the modeling of this system, we have taken the informal description that can be found in Bolt's papers. Of course, as this multimodal application has only been presented informally (using natural language), it is difficult (and impossible) to perfectly understand its functioning and more precisely the detailed integration of deictics and gestures. In order to build a formal model from these informal descriptions we have had to make some assumptions about the behavior of the system. For instance, we have supposed that the analyzing of a deictic word is at the origin of the triggering of the gesture recognition i.e. that gesture is only taken into account after a deictic is uttered.

Using the ICO formalism, we model fusion engines using two levels of abstraction. These levels of abstraction correspond to PTIC (Presentation Technique Interaction Component) and LLIC

(Logical Level Interaction Component) levels in the Arch/slinky software architecture for interactive systems (Bass 1991). Low-level events are managed by the low level part of the model. This low-level part then communicates to high-level part of the model through production of events or if needed by means of method calls. The high-level model will then communicate to the dialog part that is in charge of the global behavior of the application.



Type of Places in the Petri net:

Speech	< String >	P2	< Color >	P3	< Size >
UsrMv	< Point >	P1	< Shape >		

Figure 1: Behaviour of the low-level controller

Figure 1 presents the formal description (using the ICO formalism) of the low-level model of the fusion engine, according to the assumption presented above.

In the model, rectangles (called transitions) represent actions the system can perform while ellipses (called places) represent state variables of the system. Places can hold tokens and the distribution of tokens in the places models the current state of the system.

The initial state of the system presented in Figure 1 is modeled by one token in place UsrMv and no token in the other places. The value of the token in place *UsrMv* represents the position of user's arm (the point on the screen that he/she shows). Places and transitions are related by arcs. A transition can be fired (i.e. an action can be performed) if and only if each input place of the transition holds at least one token. When a transition is fired, one token is removed from each input place and one token is deposited in each output place. The model in Figure 1 features two other kinds of arcs. Tests arcs model the fact that a token is tested i.e. it is not removed or changed by the firing of the transition but its existence is necessary for the actual firing of the transition. Such an arc is represented between place *UsrMv* and transition *That* meaning that in order for the system to deal with the deictic word "that" (transition *That*) the system must know user's position (information contained in the token in place *UsrMv*).

Inhibitor arcs models the zero test in a Petri net. For instance, the arc between place *Speech* and transition *Voice* is an inhibitor arc (the end of the arc is a black dot) meaning that this transition can only be fired if there is no token in place *Speech* (this is the reason why transition *Voice* is shown as fireable).

Relationship between transitions and events is done by means of dedicated transitions called synchronized transitions. A synchronized transition can only be fired if it is fireable (according to the current marking of the net) and the associated event is triggered (for instance after a corresponding user action on a dedicated input device). The model in Figure 1 features only two services "Voice" and "Gesture". These services are event services as they are synchronized with user events (utterance of words and user movements).

Table 1 describes the availability function of the model in Figure 1. It describes (in row one) the fact that the event service Voice is only available when transition Voice is available in the model. In this case this is quite trivial, but it is important to note that, most of the time, a service is associated with several transitions.

Service	Transition name
Event service: Voice	{Voice}
Event service: Gesture	{Gesture}

Table 1: Availabity Function of model in Figure 1

Event Emitter	Interaction object	Event	Service
Microphone	None	Speech	Voice
Gesture Recognizer	None	Move	Gesture

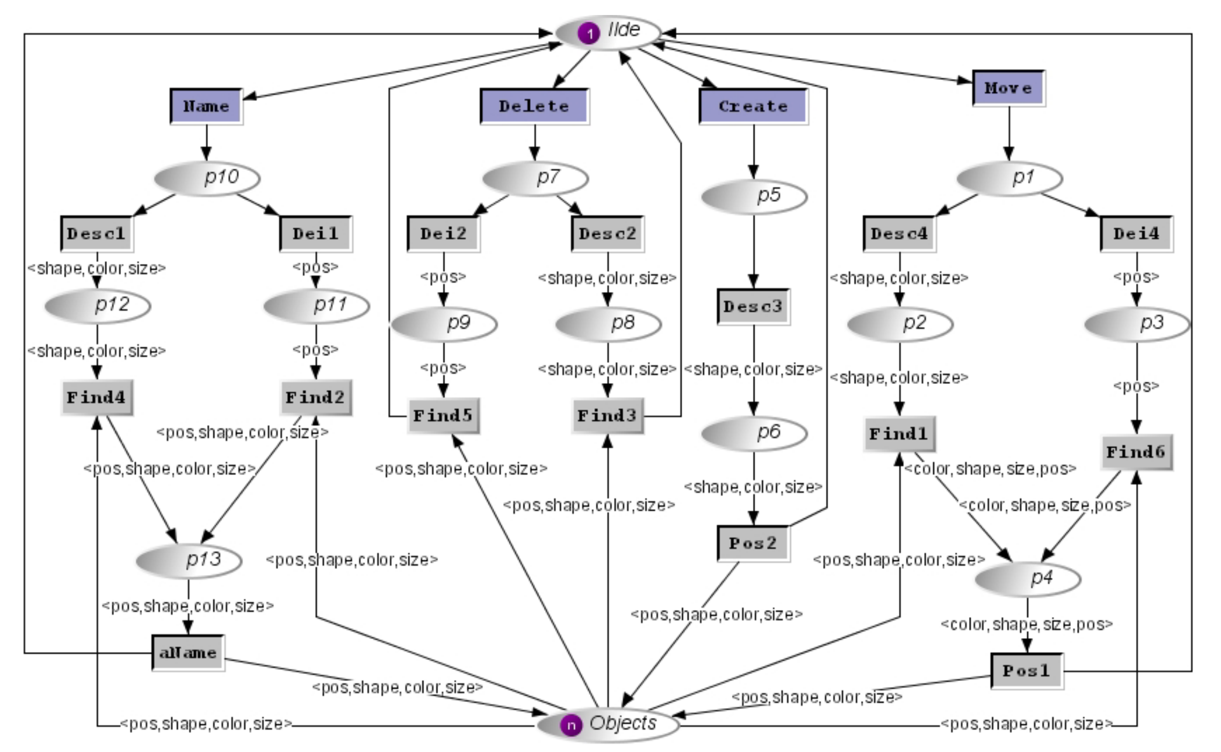
Table 2: Activation Function

Table 2 represents the correspondence between events (produced by user's actions on a media) and associated event services. For instance, this function states that if one of the transitions associated with the service Voice is available in the Petri net model and if the microphone produces a speech event, the available transition is fired.

Transition	Event produced	Transition	Event produced
Move	Move()	aName	aName(word)
Create	Create()	There	Pos(pos)
Name	Name()	That	Dei(pos)
Delete	Delete()	Object	Desc(color,shape,size)

Table 3: Event Production Function

The fusion engine presented in Figure 1 acts as a transducer: it converts an input to an output. Input events are user events (produced using *speech* or *gesture)*. Output events are higher level events to be received by the high-level model of Bolt' system (Figure 2). Table 3 describes the higher level events produced together with the transition producing them.



Type of Places:

Place p1, p5, p7, p10, Idle	< >	Place p2, p6, p8, p12	<Shape, Color, Size>
Place p3, p9, p11	<Point>	Place p4, p13, Object	<Shape, Color, Size, Point>

Figure 2: Behaviour of the high-level controller in Bolt's system

Beyond the possibility offered by the ICO of completely describing a multimodal system, formal analysis is one of benefits brought-in by the formalism. Formal analysis of the Petri net of Figure 1 guarantees that whatever state the system is in, there is always at least one transition in the model that is fireable which means that the model is live. More details about how formal analysis is performed are presented in section 4.3.

Event Service name	Transitions names	Event Service name	Transitions names
Name	{Name}	Delete	{Delete}
Make	{Make}	Move	{Move}
Create	{Create}	aName	{aName}
Desc	{Desc1, Desc2, Desc3, Desc4}		
Dei	{Dei1, Dei2, Dei4}		
Pos	{Pos1, Pos2}		

Table 4: Availabity Function of model in Figure 2

Figure 2 presents the fusion engine of Bolt's system. This model describes precisely how high-level commands are constructed according to the events produced in the low-level model in Figure 1. For instance, here is how a command such as "create a small blue rectangle here" will be processed by the model. In its initial state (one token in place *Idle* and several tokens in place *Objects*), the system is ready to produce commands. If the event "create" is produced by the low-level model, the token will move from place Idle to place P5 via the firing of transition Create. From this state, the system waits for receiving the description of the object to be created.

Event Emitter	Interaction object	Event	Service	Rendering method
Low-level model (figure 1)	None	Move	Move	None
	None	Name	Name	None
	None	Create	Create	None
	None	Delete	Delete	None
	None	Desc	Desc	None
	None	Dei	Dei	None
	None	aName	aName	None

Table 5: Activation function of model in Figure 1

This description (for instance "small blue rectangle") will be produced by the user, received by the low-level model and passed-on to the high-level model (via an event posting) and the transition Desc3 will be fired. The only information missing will be the position of the object on the screen. When the event containing this information will be received, transition Pos2 will be fired and the new object created will be stored in place *Objects* and the model will come back to its initial state (one token in place *Idle*).

Table 5 represents the correspondence between events produced by the low-level model and the services in the high-level model. Table 4 deals with the correspondence between services and synchronized transitions, i.e. which transitions in the high-level model are associated with which events.

4.3 An Example of Formal Analysis

This section shows how formal analysis techniques can be used for reasoning about MICO models. The analysis techniques used in this paper come from Petri net theory and allow us to prove properties over the fusion mechanism. Such techniques could have also been applied to other models such as input devices models or the application itself.

In Petri net theory, a set of places belong to a conservative component if and only if the number of tokens in the places remains the same whatever transitions is fired in the net. A set of transition belong to a repetitive component if and only if, given a marking M, the firing of this sequence of transition brings the net back in the marking M.

If each place of a net belong at least to a conservative component, the net is **bounded** (it does not produce nor consumes resources). If the net is **live** (it is always possible to find a sequence of firing making the firing of all transition possible), all the transitions of the net must belong to a repetitive component. Hereafter we show how such properties can be proven.

The structure of a Petri Net can be defined by : <P, T, Pre, Post> where P is the set of place of the Petri Net, T is the set of transitions ($P \cap T = \emptyset$), Pre is the input function and Post output function. The value of the element Pre (i, j) of the matrix Pre corresponds to the number of arcs from the place P_i to the transition T_j and the value of the element Post(i, j) corresponds to the number of arcs from the transition T_i to the place P_j. C is called the incidence matrix such as : C = Post-Pre. Informally C represents the consumption and the production of tokens in the net.

The conservative components are given by the solutions of the equation: $F^T.C=0$, where F is a vector corresponding to the places of the Petri net.

The repetitive components are given by the solutions of the equation: C.S=0, where S is a vector corresponding to the transitions of the Petri net.

The Petri net in **Figure 2** is defined by:

P = (Ilde, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12, p13, Objects)

T = (Name, Delete, Create, Move, aName, Desc1, Desc2, Desc3, Desc4, Dei1, Dei2, Dei4, Find1, Find2, Find3, Find4, Find5, Find6, Pos1, Pos2)

Pre and Post matrixes are not given for space reasons. Matrix C=Post-Pre (lines are the places and columns the transitions) is given in Table 6.

-1	0	0	0	-1	0	0	0	0	0	0	1	0	1	0	-1	-1	1	1	1
0	0	0	0	0	0	0	0	0	-1	-1	-1	-1	-1	-1	0	0	1	1	1
0	0	0	-1	0	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0
0	-1	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	-1
0	0	0	0	0	0	0	0	1	-1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
1	0	0	0	0	0	0	-1	0	1	0	0	0	0	1	0	0	-1	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	-1	0	0	0	0	0	0	0	0	0	0	0	-1	0
0	0	-1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0

Table 6: C matrix of the model in Figure 2

The calculus of conservative and repetitive component can be done automatically, so we only give here the solutions of these equations. In PetShop, the tool that supports the basic ICO formalism (but not the extensions presented in this paper) such calculus are made in background providing designers with analysis results while editing the models (Navarre et al. 2003).

The set of conservative component of the net is:

Ilde + p1 + p2 + p3 + p4 + p5 + p6 + p7 + p8 + p9 + p10 + p11 + p12 + p13

And the repetitive components are:

Move + Desc4 + Find1 + Pos1; Move + Dei4 + Find6 + Pos1; Name + Dei1 + Find2 + aName; Name + Desc1 + Find4 + aName; Create + Desc3 + Pos2 + Delete + Desc2 + Find3 et Create + Desc3 + Pos2 + Delete + Dei2 + Find5

From that analysis we can see that all places are in the same conservative component (the net is bounded and the number of tokens in this set of places remains the same at all times) and each transition is at least in a repetitive component (which is a necessary condition for the net to be live).

These analysis results are useful for understanding the behavior of the system in terms of more abstract properties. For instance, the fact that the net is live guarantees that the fusion mechanism is deadlock free i.e. it is always able to process input and to produce events. Using other analysis techniques such as marking graph calculation (not presented here) it is also possible to prove that the fusion mechanism is reinitialisable i.e. whatever events are received; it is always possible to come back to the initial state. This is very important to ensure that events received at some point do not alter the behavior of the fusion mechanism for future activities. More information about such properties can be found in (Accot et al. 1997).

5 Conclusion and future work

The continuously increasing complexity of the information manipulated by interactive systems calls for new interaction techniques in order to increase the bandwidth between the system and the user. Multimodal interaction techniques are considered as a promising way for tackling this problem. However, the lack of engineering techniques and processes for such systems makes them hard to design and to build and thus jeopardizes their actual exploitation in the area of safety critical application.

This paper has presented a formal description technique that can be used for the modeling and the analysis of multimodal interactive systems. This work is part of a project on the evaluation and use of multimodal interaction techniques in the field of command and control real time military systems.

The paper has presented extensions to the ICO formalism that is dedicated to the design specification, verification and prototyping of interactive systems. Its formal underpinnings make it especially suitable for safety critical interactive systems. ICO formalism has been applied to various kinds of systems including business, Air Traffic Management and command and control applications.

In this paper we have emphasized the use of the formalism for the description and verification of fusion mechanism for input events. We are currently integrating the extensions presented here into PetShop the case tool supporting the edition, verification and interactive prototyping of ICO models (Bastide et al., 2002).

6 Acknowledgments

The work presented in the paper is partly funded by French DGA under contract #00.70.624.00.470.75.96 and EU via the ADVISES Research Training Network RTN2-2001-00053. Special thanks are due to Rémi Bastide and David Navarre for fruitful discussions about this work.

7 References

Accot, J. Chatty, S. and Palanque, P. (1996), A Formal Description of Low Level Interaction and its Application to Multimodal Interactive Systems, 3rd Eurographics workshop on Design, Specification and Verification of Interactive systems, Springer-Verlag, pp. 92-104.

Accot, J. Chatty, S. Maury S. and Palanque, P. (1997) Formal Transducers: Models of Devices and Building Bricks for Highly Interactive Systems 4th Eurographics workshop on "design, specification and verification of Interactive systems", Springer Verlag, pp. 143- 160.

Bass, Len, R., Pellegrino, R. Reed, S., Seacord, R., Sheppard, et Szezur, M. R. (1991), The Arch model: Seeheim revisited User Interface Developpers'workshop, version 1.0.

Bastide R. and Palanque P. (1995), A Petri Net Based Environment for the Design of Event-Driven Interfaces, *in* Lecture Notes in Computer Science n° 935 - *ATPN'95: proceedings of the 16th International Conference on Application and theory of Petri Net,* Springer Verlag.

Bastide R., Navarre D. and Palanque P. (2002), A Model-Based Tool for Interactive Prototyping of Highly Interactive Applications, *Proceedings of the ACM SIGCHI 2002 (Extended Abstracts)*, pp. 516-517.

Bastide R., Palanque P., Le Duc H., Muñoz J. (1998), Integrating Rendering Specifications into a Formalism for the Design of Interactive Systems, *DSV-IS'98: proceedings of the 5th Eurographics workshop on Design, Specification and Verification of Interactive systems*, Springer Verlag.

Bastide R. and Palanque P. (1994), Petri Net based design of user-driven interfaces using the Interactive Cooperative Objects formalism, *DSV-IS'94: Interactive systems: design, specification, and verification*, Springer-Verlag, pp. 383-400.

Bederson B. B. Meyer J. and Good L. (2000). Jazz an Extensible Zoomable User Interface Graphics Toolkit in Java, *UIST'2000, ACM Symposium on User Interface Software and Technology*.

Bier, E. Stone, M. Pier, K. Buxton, W. and DeRose, T. (1993), Toolglass and Magic Lenses: the See-Through Interface, *Proceedings of ACM SIGGRAPH'93*, ACM Press, pp. 73-80.

Bolt, R and Herranz, E. (1992), Two-Handed Gesture in Multi-Modal Natural Dialog, P*roceedings of the fifth annual ACM symposium on User interface software and technology*, ACM Press, p 7-14.

Bolt, R. (1980), Put That There: Voice and Gesture at the Graphics Interface, *SIGGRAPH'80*, p262-270.

Buxton, W. and Myers, B. (1986), A Study in Two-Handed Input, *Proceeding of the ACM CHI,* Addison-Wesley, pp. 321-326.

Chatty, S. (1994), Extending a Graphical Toolkit for Two-Handed Interaction, *Proceedings of the ACM symposium on User Interface Software and Technology,* ACM Press, pp. 195-204.

Cohen, P. Johnston, M. McGee, D. Oviatt, S. Pittman, J. Smith, I. Chen, L. and Clow, J. (1997), QuickSet: Multimodal Interaction for Distributed Applications, *Proceedings of the fifth ACM Int. conference on Multimedia*, ACM Press.

Coutaz, J. and Nigay L. (1993), A Design Space for Multimodal Systems Concurrent Processing and Data Fusion, *Human Factors in Computing Systems, INTERCHI'93*, pp. 172-178.

Coutaz, J. Nigay, L. Salber, D. Blandford, A. May, J. and Young, R. (1995), Four Easy Pieces for Assessing the Usability of Multimodal in Interaction the CARE Properties, *Human Computer Interaction, Interact' 95*, pp. 115-120.

Coutaz, J. Salber, D. Carraux, E. and Portolan, N. (1996), Neimo, a Multiworkstation Usability Lab for Observing and Analysing Multimodal Interaction, *Human Factors In Computing Systems CHI'96 Conference Companion*, ACM Press, pp. 402-403.

Duke, D. and Harrison, M. D. (1997), Mapping User Requirements to Implementations, *Software Engineering Journal*, Vol 10(1), pp. 54-75.

Genrich, H. J. (1991), Predicate/Transition Nets, *in K.* Jensen & G. Rozenberg (eds.), *High-Level Petri Nets: Theory and Application,* Springer Verlag, pp. 3-43.

Hinckley K., Czerwinski M. and Sinclair M. (1998), Interaction and Modelling Techniques for Desktop Two-Handed Input, *Proceedings of ACM UIST*, ACM Press, pp. 49-58.

MacColl & Carrington, D. (1998), Testing MATIS: a Case Study On Specification-Based Testing of Interactive Systems, FAHCI, pp.57-69, ISBN 0-86339-7948.

Navarre David, Palanque Philippe & Bastide Rémi. A Tool-Supported Design Framework for Safety Critical Interactive Systems in Interacting with computers, Elsevier, to appear 2003.

Nigay, L. and Coutaz, J. (1995), A Generic Platform for Addressing the Multimodal Challenge, Proceedings on Human factors in computing systems, pp. 98-105.

Nigay, L. and Vernier, F. (2000), A Framework for the Combination and Characterization of Output Modalities, *in* P. Palanque, F. Paterno (eds), p*roceeding of the DSV-IS'2000 Conference,* Springer- Verlag, pp. 35-50.

Parnas, D. L. (1969), On the Use of Transition Diagram in the Design of a User Interface for Interactive Computer System, *Proceedings of the 24th ACM Conference,* ACM Press.

Vo, M.T. and Wood, C. (1996), Building an Application Framework for Speech and Pen Input Integration in Multimodal Learging Interface, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol 6, pp. 3545-354.