

Generic Multi-Agent Architectures for Multimedia Multimodal Dialogs

H. DJENIDI⁽¹⁾, A. RAMDANE-CHERIF⁽²⁾, Pr. C. TADJ⁽¹⁾ and Pr. N. LEVY⁽²⁾

⁽¹⁾*Electrical Engineering Department, École de Technologie Supérieure
1100, Notre-Dame Ouest, H3C 1K3, Montreal, Quebec, Canada.
{hdjenidi,ctadj}@ele.etsmtl.ca*

⁽²⁾*PRiSM, Université de Versailles St.-Quentin,
45, Avenue des Etats-Unis, 78035 Versailles Cedex, France
{rca, nlevy}@prism.uvsq.fr*

Abstract: The multimodal featuring fusion for natural human-computer interaction involves complex intelligent architectures facing unexpected errors and mistakes made by users. These architectures should react to events that occur simultaneously with eventual redundancy from different input media. In this paper, intelligent agent based generic architectures for multimedia multimodal dialog protocols are proposed. Global agents are decomposed into their relevant components. Each element is modeled separately using timed Colored Petri networks. The elementary models are then linked together to obtain the full architecture. Hence, maintainability, understandability and the modification of the architecture are facilitated. For validation purpose, the proposed multi-agent architectures are applied on a practical example.

Keywords: Multimedia multimodal dialog fusion, Multi-agent architecture, Timed Colored Petri networks.

1 Introduction

With the growing technology, many applications supporting more transparent and flexible human computer interactions have emerged. This results in an increasing need for more powerful communication protocols, especially when several media are involved. Multimedia multimodal applications mean systems combining natural input modes, such as speech, touch, manual gestures, etc. Thus, a comprehensive command or a meta-message is generated by the system and sent to multimedia output devices. A system-centered definition of multimodality is used in this paper. The multimodality conveys two striking features that are relevant to the software design of multimodal systems:

- the fusion of different types of data from different Input devices, and
- the temporal constraints imposed on information processing from/to Input/Output devices.

Since the first rudimentary but pertinent system, "Put That There" [1], which processes speech in parallel with manual pointing, different multimodal applications have been developed [2, 3, 4]. Each application is based on a dialog architecture combining modalities to match and elaborate on the relevant multimodal information. In such elaborations, projects usually begin from scratch and are generally based exclusively on the experiences of the designers. Consequently, they remain replications of previous results and limited synergy among parallel ongoing efforts. Today, there is no agreement on generic architectures that reflects a dialog implementation, independently of the application type. The main objective of this paper is to propose a generic architecture to analyze and extract the collective and recurrent properties, implicitly used in such dialogs.

This paper presents architectural paradigms for multimedia multimodal fusion. These paradigms use the agent architectural concept to achieve their functionalities and unify them

into generic structures. The modular structure of the proposed architecture allows easy monitoring of the global template.

The next Section summarizes the real time requirements of such dialog architectures for multimedia multimodal applications. Section 3 presents generic multi-agent architectures based on the previous analysis. Section 4 discusses the use of timed Colored Petri Networks (CPN) to model such architecture. Section 5 illustrates the proposed architecture with a stochastic timed CPN [5, 6] example of the classical "Copy and Paste" operations. Simulation tests are processed using Design CPN Tool Kit [7].

2 Multimodal Dialog Architectures: Overview and Requirements

With the increasing complexity of multimedia applications, a single modality becomes insufficient to allow the user to interact effectively across environments. A basic multimedia multimodal system as shown in Figure 1, offers the user the possibility to decide which modality or combination of modalities are better suited, depending on the task and environment contexts (see examples in [8, 9]).

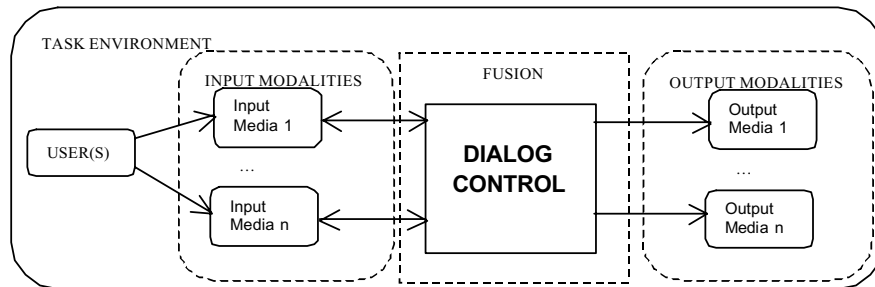


Fig. 1. Basic multimedia multimodal model (\leftrightarrow : interaction, \rightarrow : action).

The environmental conditions could lead to more constrained architectures that have to be adaptable during the continuous change of either external perturbations or the user's actions. In this context a first framework is introduced in [10] to classify interactions. It considers two dimensions ('engagement' and 'distance') and decomposes the user/system dialog into four types.

Engagement	Distance	Type of System
Conversation	Small	high-level language
Conversation	Large	low-level language
model world	Small	direct manipulation
model world	Large	low-level world

Table 1. Interaction Systems.

The 'engagement' characterizes the depth implication of the user in the system. The user feels that an intermediary subsystem performs the task, in 'conversation' case, and that he can act directly on the system components in 'model world' case. The 'distance' represents the user cognitive effort taken.

This framework reaches the idea that two kinds of multimodal architectures are possible [11]. The first one makes fusions based on feature signal recognition. The recognition process steps of one modality guide and influence the other modalities in their own recognition steps [12, 13]. The second architecture uses individual recognition systems for each modality. Such systems are associated with an extra process that performs semantic fusion of the individual recognized

signal elements [1, 3, 14]. A third hybrid architecture is possible by mixing the two previous types: signal feature level and semantic information level.

At the core of multimodal system design, the information fusion of the input modes is the main challenge. The input modes can be equivalent, complementary, specialized or redundant as described in [15]. In this context, the multimodal system designed with one of the previous architectures (features or/and semantic levels) needs the integration of the temporal information. Figure 2 (left) shows the possible type of multimodality depending on the time proximity of the input signals. Time granularity is an important decision criterion when we generate a multimodal semantic sequence as shown in Figure 2 (right). In this example, it shows that the chosen multimodality type, for mouse clicks and speech, is the synergistic one. This is obvious in the example, because the click occurs only during the time when a sentence is said. The synergistic mouse/speech actions correspond to one statement and the tactile screen action to another one. Both statements are performed in parallel and could be independent, equivalent, complementary, specialized and/or redundant.

In other words, the temporal aspect in multimodal architecture does not only handle signals overlapping. It helps to decide whether two signal parts should belong to a multimodal fusion set or whether they should be considered as separate modal actions. Therefore, multimodal architectures are better able to avoid and recover errors that mono-modal recognition systems can't recover [14, 11]. This property results in a more robust, natural, human-machine language. Another property is that, the more timed combinations of signal information or semantic multi-signal grow, the more equivalent formulations of the same command are possible. For example, ["Copy that there"], ["copy" (click) "there"] and ["copy that" (click)] are various ways to represent three statements of a same command (copying a object in place), if speech and mouse clicking are used. This redundancy also increases the robustness in terms of error interpretation.

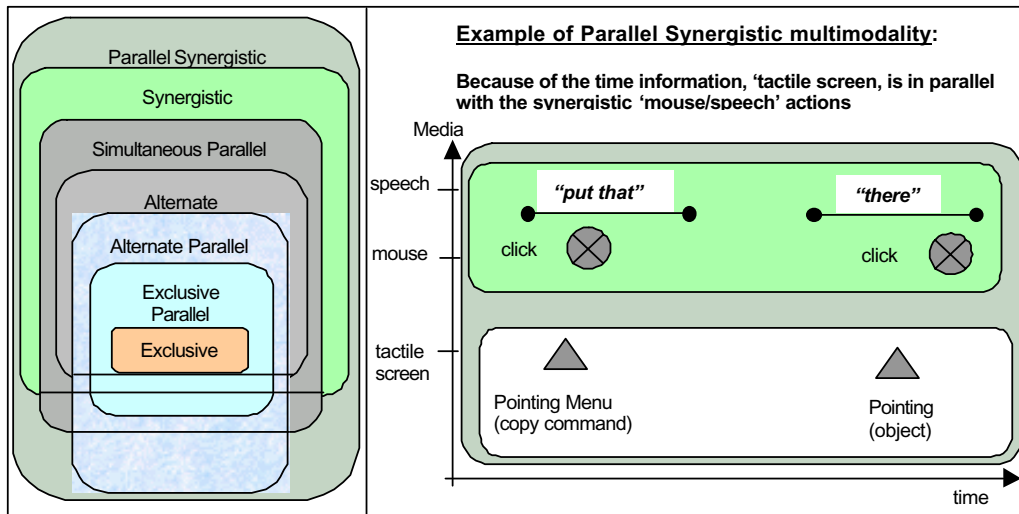


Fig. 2. Inclusion relations between types of multimodality (left), and a three timed media example (right) [25].

Figure 3 summarizes the main requirements and characteristics needed in multimodal dialog architectures. As shown in this figure, five characteristics can be used in the two different levels of the fusion operations: the 'early fusion' at the feature fragments level and the 'late fusion' at the semantic one [11].

The Asynchronous property gives the architecture the flexibility to handle multi external events while parallel fusions are still processing. The specialized fusion operation deals with an attribution of a same modality to a same statement type. (For example, in drawing applications, speech is specialized for color statements and pointing for basic shape statements.)

The granularity of the semantic and statistic knowledge depends on the media nature of each input modality. This knowledge leads to important functionalities. It lets the system accept or refuse the multi input information for several possible fusions (selection process); and it helps the architecture choose, between several fusions, the most suitable command to execute or message to send to an output media (decision process).

The property of parallelism is, obviously, inherent to such applications involving multi inputs.

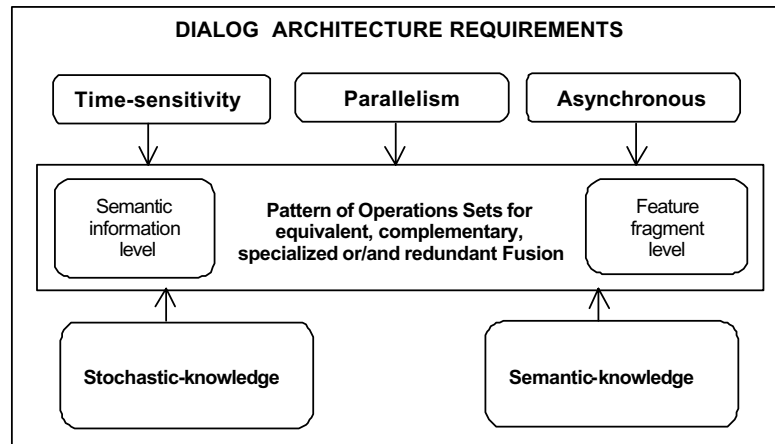


Fig. 3. The main requirements for multimodal dialog architecture (→: used by).

The whole requirements suggest strongly intelligent multi-agent architectures, which are the purpose of the next section.

3 Generic Multi-Agent Architectures

The Agents are entities that can interact and collaborate with dynamic and synergy for modality combination issues. The interactions should occur between agents and agents should also get information from users. An intelligent agent has three properties. It reacts in its environment at certain times (reactivity), takes initiatives (pro-activity) and interacts with other intelligent agents or users (sociability) to reach goals [16, 17, 18]. Therefore each agent could have several input ports to receive messages and/or several output ports to send ones.

The level of intelligence of each agent varies according to two major options coexisting today in the field of Distributed Artificial Intelligence [19, 20, 21]. The first one, corresponding to the cognitive school, attributes the level to the cooperation of very complex agents. This approach deals with agents with strong granularity assimilated to expert systems.

In the second school the agents are simpler and less intelligent but more active. This reactive school presupposes that it is not necessary to each agent to be individually intelligent to reach an intelligent total behavior [22]. This approach deals with a cooperative team of working agents with low granularity, which can be matched to finite automate.

Both approaches can be matched to the late and early fusions of multimedia multimodal architectures.

Obviously, there are all the possible intermediaries between these options of multi-agent systems. One can easily imagine systems based on a modular approach, putting sub-modules in competition, each sub-module being itself a universe of overlapping components. This word is usually employed for 'sub-agents'.

Identifying the generic parts of multimodal multimedia applications and binding them into an intelligent agent architecture require the determination of common and recurrent communication protocols and their hierarchical and modular properties in such applications.

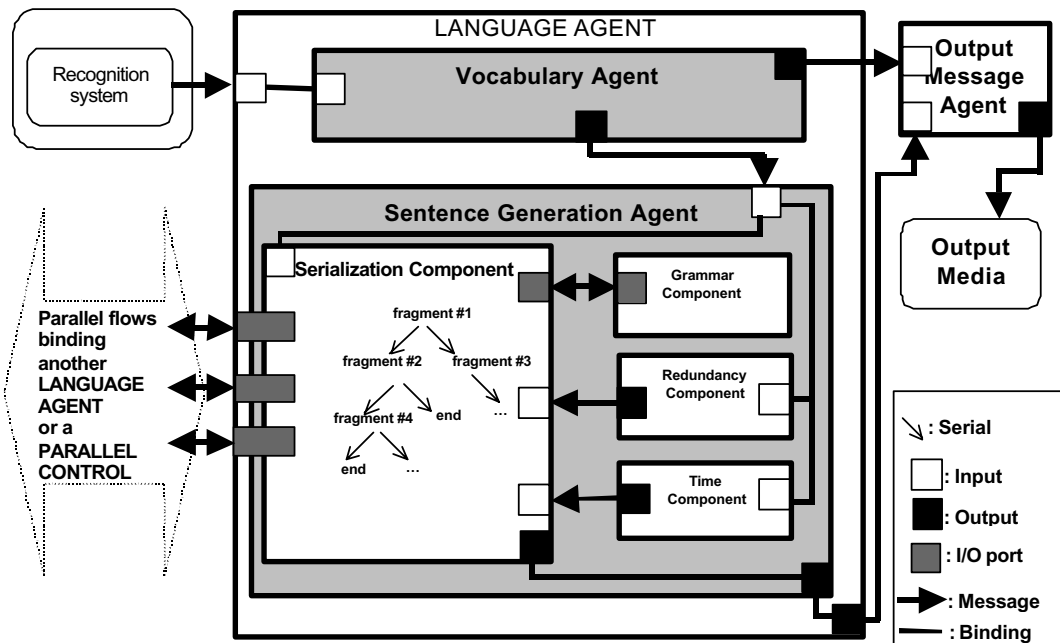


Fig. 4. Generic Language Agent corresponding to an input modality.

In most multimodal applications, the speech, as input modality, offers speed, a large information spectrum and relative facility of use. It lets both the user's hands and eyes free to work in other necessary tasks present, for example, in driving or moving cases. More over, speech involves a generic communication language pattern between the user and the system.

This pattern is described by a grammar with production rules, able to serialize possible sequences of the vocabulary symbols produced by users. The vocabulary could be word set, phoneme set or another signal fragment set depending on the feature level of the recognition system. The goal of the recognition system is to identify signal fragments. Then, an agent organizes the fragments in a serial sequence according to his grammar knowledge and asks others agents for possible fusion at each step of the serial regrouping. The whole interaction can be synthesized in a first generic agent architecture, as shown in Figure 4, called Language Agent (LA).

Each input modality must be associated with an LA. For basic modalities like manual pointing or mouse clicking, the complexity of the LA is strongly reduced. The 'Vocabulary Agent' that checks whether or not the fragment is known, is, obviously, no longer necessary. The 'Sentence Generation Agent' is also reduced into a simple event thread whereon another external control agent could possibly make parallel fusions. In such a case, the external agent could handle 'Redundancy' and 'Time' information, with to corresponding components. These two components are agents that, respectively, check redundancies and time neighborhood of the fragments during their sequential regrouping (Figure 4). The 'Serialization Component' processes this regrouping. Thus, depending on the input modality type, the LA could be assimilated to an expert system or to a simple thread component.

Two or more LAs can communicate directly for early parallel fusions or, through another central Agent, for late ones (Figure 5). This central agent is called Parallel Control Agent.

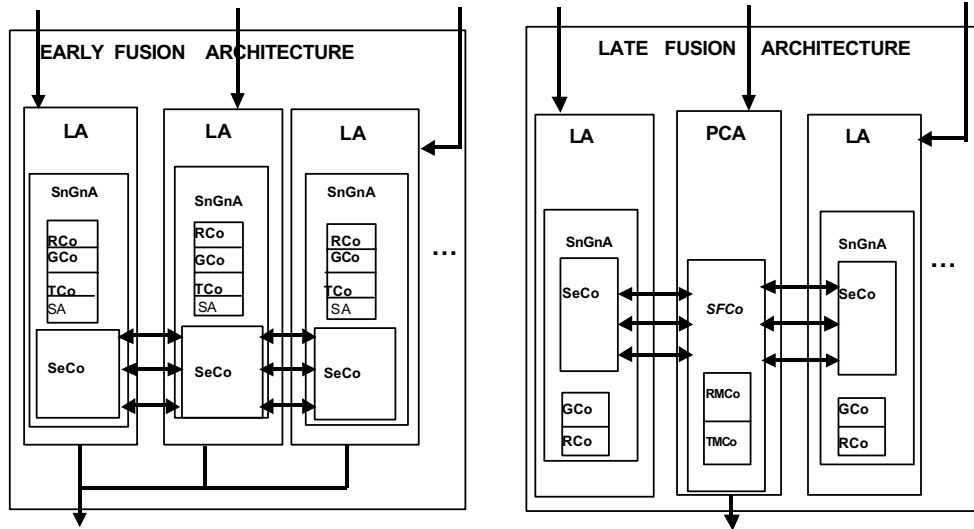


Fig. 5. Principles of early and late fusions architectures (L: language, P: parallel, C: control, A: agent, G: grammar, S: semantic, Sn: sentence, Gn: generation, F: fusion, Se: serialization, Co: component, T: time, R: redundancy and M: management). More connections (arrows) could be added or removed by the agents to gather fusion information.

In the first case, the ‘Grammar Component’ of one of the LAs must carry an extra semantic knowledge for the parallel fusion purpose. This knowledge could also be distributed between the LA’s ‘Grammar Components’, as shown in Figure 5 (left). Several Serializing Components share their common information until one of them gives the sequential parallel fusion output. In

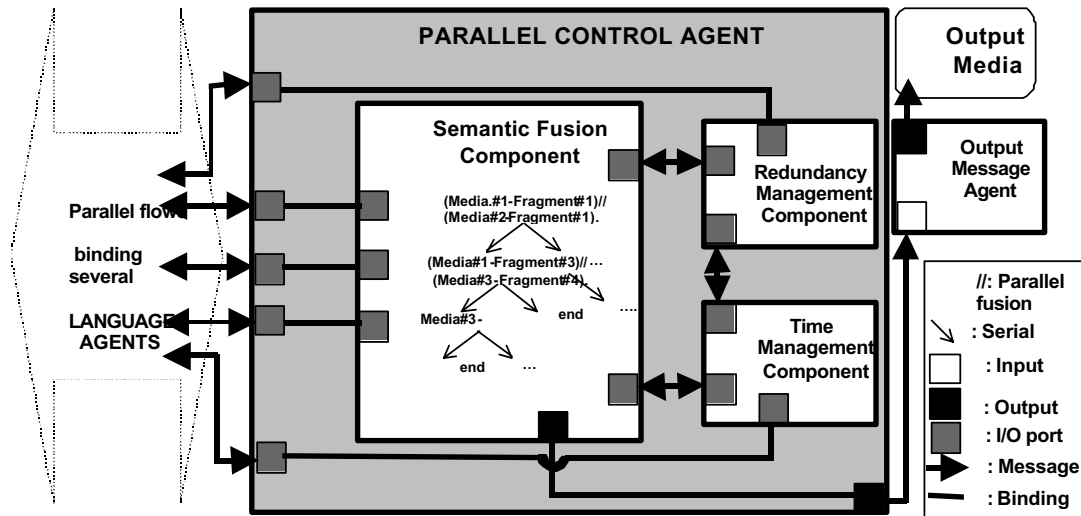


Fig. 6. Generic Parallel Control Agent for central parallel multimodal fusions.

the other case (Figure 5 right), a ‘Parallel Control Agent’ (PCA) handles and centralizes the parallel fusions of different LA information. For this purpose, the PCA has two intelligent components for, respectively, Redundancy and Time managements (Figure 6). These agents exchange information with other components to elaborate the decision. Then, generated authorizations are sent to the Semantic Fusion Component (SFCo). Based on these agreements,

the SFCo carries the steps of the semantic fusion process. As shown in Figure 6, Redundancy and Time Management Components receive the redundancy and time information via the Semantic Fusion Component or directly from the LA, depending on the complexity of the architecture and the designer choices.

The paradigms proposed in this section constitute an important step in the software development of multimodal user interface (Figure 7). Another important phase of the software development, for such

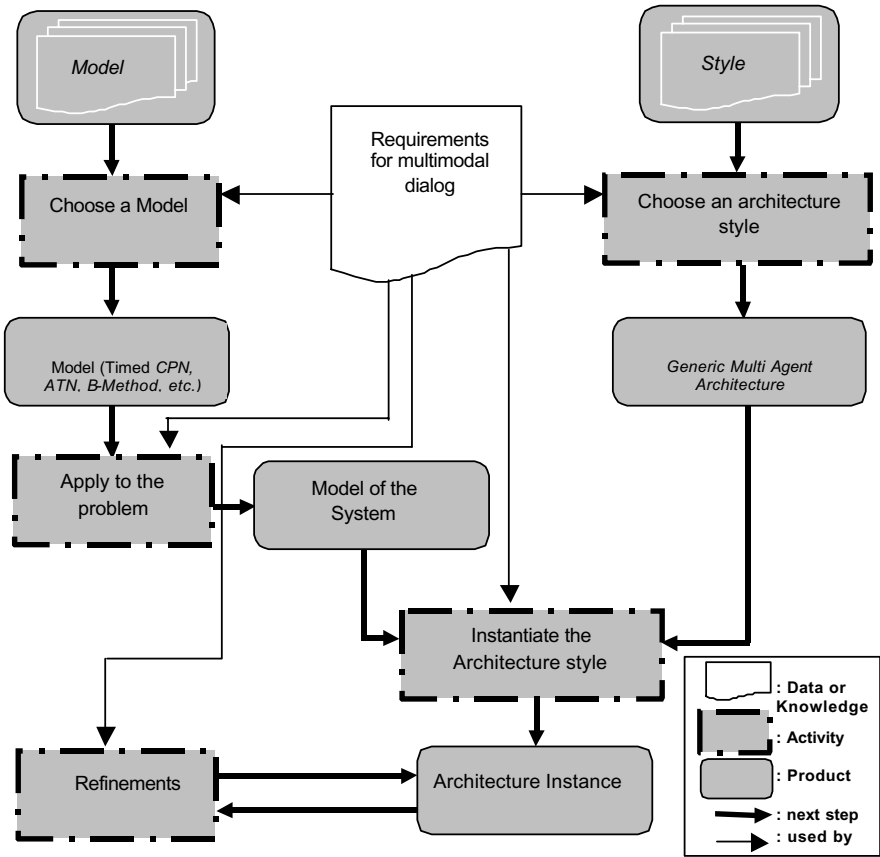


Fig. 7. The software development phases of multimedia multimodal dialog architecture. (ATN: Augmented Transition Nets)

applications, concerns the modeling aspect. Different methods like B_method [24], Augmented Transition Networks [25], or Timed CPN [6, 7], can be used to model the multi-agent dialog architectures. The next Section discusses the choice of Colored Petri Networks to model these architectures.

4 Petri Nets Modeling

Recently small augmented finite-state machines like augmented transitions networks have been used in the multimodal presentations system [26]. These networks easily conceptualize the communication syntax between input and/or output media streams. However, they have limitations when important constraints such as temporal information and stochastic

behaviors need to be modeled in protocols of fusion. Timed Stochastic Colored Petri Networks offer a more suitable pattern [5, 6, 7] to design such constraints in multimodal dialog.

For modeling purpose, each input modality is assimilated to a thread where signal fragments flow. Multimodal inputs are parallel threads corresponding to a changing environment that describes different internal states of the system. Multi-agent systems are multi-threaded: each agent has a control on one or several threads. Intelligent agents observe the states of one or several threads for which it is designed. Then, the agents execute actions that modify the environment. In a more formal way [17],

$$\text{if } \mathbf{A} = \{ \mathbf{a}_1; \mathbf{a}_2; \dots \} \quad (1),$$

$$\text{and } \mathbf{O} = \{ \mathbf{o}_1; \mathbf{o}_2; \dots \} \quad (2),$$

are the sets of actions and observations of an agent, respectively

$$\text{and if, } \mathbf{S} = \{ \mathbf{s}_1; \mathbf{s}_2; \dots \} \quad (3),$$

is the set of states with which the environment is described (including intermediary states), then the Petri network models two kind of activities described by the functions

$$\mathbf{Observation_function} : \mathbf{S} \rightarrow \mathbf{O} \quad (4),$$

$$\mathbf{Environment_function} : \mathbf{S} \times \mathbf{A} \rightarrow 2^{\mathbf{S}} \quad (5).$$

The first function describes what an agent observes, in a certain state \mathbf{s}_i . The second one describes how the environment develop the state \mathbf{s}_i when an action \mathbf{a}_i is executed. The Petri network models also the actions of the agents described by the function

$$\mathbf{Action_function} : \mathbf{O} \rightarrow \mathbf{A} \quad (6).$$

The characteristic behavior of an agent action in an environment is the set 'History':

$$\mathbf{History} = \{ \mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_i, \dots \} \quad (7)$$

of all sequences of the observations defined by

$$\mathbf{h}_i: (\mathbf{s}_0) \xrightarrow{\mathbf{a}_0} (\mathbf{s}_1) \xrightarrow{\mathbf{a}_1} \dots (\mathbf{s}_i) \xrightarrow{\mathbf{a}_i} \dots \quad (8)$$

$$\text{with } \mathbf{a}_i = \mathbf{Action_function} (\langle \mathbf{s}_0, \dots, \mathbf{s}_i \rangle), \quad \forall i \quad (9)$$

$$\text{and } \mathbf{s}_i = \mathbf{Environment_function} (\mathbf{s}_{i-1}, \mathbf{a}_{i-1}), \quad \forall i, i \neq 0 \quad (10)$$

(\mathbf{s}_0 is the initial state of the system).

To summarize the precedent transaction, the Petri network has to model the functions (4), (5), (6) and also the input media threads with the design CPN toolkit [7]. In the following, it is assumed that this toolkit and its semantics are known. The Petri network is a diagram flow of interconnected places (or locations represented by ellipses) and transitions (represented by boxes). Labeled arcs connect places to transitions. The CPN is managed by a set of rules. The rules determine when an activity can occur and specify how its occurrence changes the state of the places by changing their colored marks. The set of colored marks in all places before an occurrence of the CPN is equivalent to an observation sequence of a multi-agent system. Each

mark is a symbol that could represent signal fragments (pronounced words, mouse clicks, etc.), serialized or associated fragments (comprehensive sentences or commands) or simply a variable. In CPN each mark can be of all the data types generally available in a computer language: integer, real, string, Boolean, list, record and so on. These types are called colorsets. Thus, a CPN is a graphical structure with associated computer language statements. A transition represented by a box can model an agent. The observation function of an agent is simply modeled by input arc inscriptions and also by the conditions in each transition guard (symbolized by **[conditions]** under a box in the example represented figures 12). Input arc inscriptions specify data that must exist for an activity to occur. When a transition is fired (an activity occurs), a mark is removed from an input place and the transition activity modifies the data associated to the mark and thereby changes the state of the system (by adding a mark in an output place). If there are colorset modifications to perform, they are executed by a program associated to the transition (The program is written in a dashed line box close to the concerned transition as shown in figures 12 (a) and (b) and the symbol **[C]** specifies that a code is attached to the transition). Also, output arc inscriptions specify data that will be produced if an activity occurs. Thus, CPN provide an extremely effective dynamic modeling paradigm. In summary, the set of colored marks in all places before an occurrence of the net is equivalent to an observation sequence of a multi-agent system. A transition represented by a box can model an agent as shown in the examples of Figure 8.

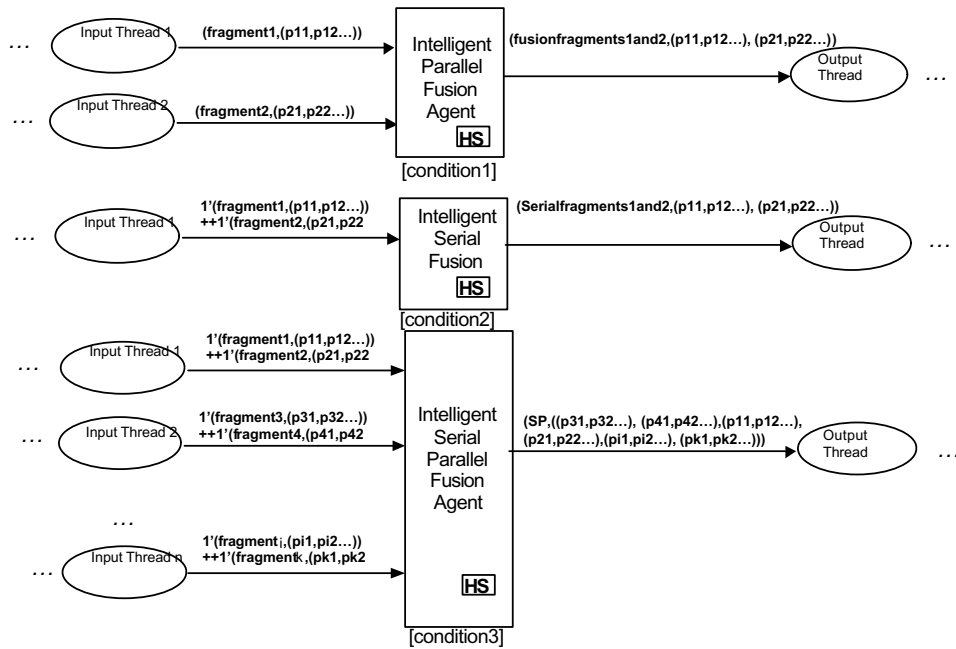


Fig. 8. Principle of parallel, serial and serial→parallel fusions modeled by Petri Nets.

In Figure 8, the variables, like ‘p11’, ‘p12’, etc (beginning with the character ‘p’), are used to represent the time, grammatical and semantic informations of the signal fragments. The next section shows the use of variables in a practical example. The observation function of an agent is simply modeled by the conditions in each transition guard. The transition activity modifies data and thereby changes the states of the system.

5 Simulation example

In this section a typical example of a distributed architecture is presented. The ‘Copy and Paste’ architecture chosen involves a high level LA, for speech modality, linked, by a PCA, to a rudimentary mouse clicking LA. Tables 2 and 3 give the vocabulary, used by the speech LA, and the basic corresponding grammar. Each word has a label used in the network design.

Word	Word label
open	1
close	2
delete	3
copy	4
paste	5
cancel	6
that	7

Table 2. Vocabulary

Symbolic regular expressions are used to represent semantic elements. These expressions use the arrow operator for sequential concatenation in the time domain. For example, in the semantic expression:

(word 1→word 2)

word 1 is simply followed by (or contiguous to) word 2. In the following table, the codes (last column) are simply obtained by summing the word labels of each semantic code. The obtained codes give information used by the speech LA for serial constructions of sentences.

Set of Sentences	Command meaning	Set of corresponding semantic codes	Set of corresponding codes
{ (open→that); (open) }	Open object	{ (1→7); (1) }	{ (8); (1) }
{ (close→that); (close) }	Close object	{ (2→7); (2) }	{ (9); (2) }
{ (delete→that); (delete). }	Delete object	{ (3→7); (3) }	{ (10); (3) }
{ (paste) }	Past last copied object	{ (5) }	{ (5) }
{ (copy→that); (copy) }	Copy object	{ (4→7); (4) }	{ (11); (4) }
{ (cancel) }	Cancel last command	{ (6) }	{ (6) }

Table 3. Grammar of authorized sentences.

The word ‘cancel’ is a command that automatically cancels the last action among the authorized sentences. Therefore, if the user says one of the words labeled in the set {1, 2, 3, 4, 5} just after “cancel”, the time proximity between the two words is the decision criterion for suppressing the second word or taking it as a next command. For the proposed architecture both scenarios are processed. Non-authorized sentences used in this architecture are given in Table 4.

Non authorized Sentences	Corresponding semantic codes	Corresponding codes
(that), (that→...)	(7), (7→...)	(7), (8)...(13)
(paste→that)	(5→7)	(12)

Table 4. Grammar of non-authorized sentences and their codes.

The multimodal dialog gives for each sentence a set of possible redundant fusions. The symbol // models these concurrent associations in regular expressions. For example, depending upon

temporal information, the first command given in Table 3 is an element of the following semantic fusion set:

{(click→open→that); (open→click); (click→open); (click // open); ((click // open)→that); (click//(open→that))}.

This semantic set includes the grammatical sentences corresponding to the command ‘Open object’. Words temporally isolated and labeled in the set **{1, 2, 3, 4, 7}** are not considered by the PCA. The remaining fusion entities like **((close→open) // click), (click // (delete→open))**, etc. or isolated clicks are also ignored by the system. The whole sets constitute the semantic knowledge. The main focus in this paper is how to use a timed semantic knowledge to achieve a multimodal fusion. The global declaration page, used in the timed CPN example, is shown in Figure 9 below.

```

(*GLOBAL DECLARATION PAGE*)
(*=====*)
(*Proximity time between two events*)
(*=====*)
val ProxyTime = 100;
(*Average Inter_arrival*)
(*=====*)
val ClickArrival = ref 1.0;
val WordArrival = ref 10.0;
(* Color sets *)
(*=====*)
color Int =int;
color Attribute = product Int * Int * Int;
(* Color sets for mouse event *)
(*****)
color MouseClick = with ClickEvent;
color ClickxAttribute = product MouseClick * Attribute timed;
color ME = with me timed;
(* Color sets for speech*)
(*****)
color WordSaid = with Word;
color WordxAttribute = product WordSaid * Attribute * Int timed;
color WE = with we timed;
(* Color sets for fusion event*)
(*****)
color FusedEvents = with Fused;
color FAttrib =product Int * Int;
color FusedxAttribute = product FusedEvents * FAttrib * FAttrib * FAttrib timed ;
(*Variables*)
(*=====*)
var Word1, Word2,Word3 :WordSaid;
var n, Fn, Fm, Fm1, Fm2, Fm3, m, m1, m2, m3, p, Fp : Int;
(* Variables for Time *)
(*****)
var NextClick, NextWord, ArrivalTimeC, ArrivalTimeW1, ArrivalTimeW2,
ArrivalTimeW3, ArrivalTimeWp, ArrivalTimeW: Int ;
(* Variables for word labels *)
(*****)
var wt, wtype, wtype1, wtype2, wtype3 : Int;
(* Functions *)
(*=====*)
fun intTime()=IntInf.toInt(time());
fun round x =floor(x+0.5);
fun ExpLaw x= round(exponential(x));

```

Fig. 9. Global declaration page.

Also, a place filled with a pattern symbolizes a thread with input and output ports, as shown in Figure 10.

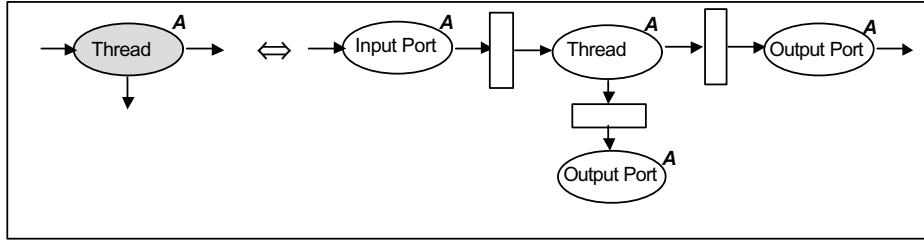


Fig. 10. Symbolic representation used in figures 12 (a) and (b).

Input ($\boxed{P} \boxed{In}$) and output ($\boxed{P} \boxed{Out}$) ports of the Petri network correspond as well to the architecture ones (see Figures 4 and 6). The associated network, as shown in Figures 12 (a) and (b), uses two random generators to design the arrival time of the input media events. The generators are drawn in the top of Figure 12 (a) with dashed non-bold lines and both are modeled with the transitions named 'Click' and 'WordSaid'. The inter-arrival time between two pronounced words as well as the time between two consecutive 'clicks', are exponentially distributed. Events (like words and clicks) are generated or arrived in two different threads (the places 'ThreadofClick' and 'ThreadofWords'). The time between two click (respectively word) arrivals has a mean = ClickArrival (respectively = WordArrival). The inter-arrival time between 2 click (respectively word) events has an exponential distribution with parameter $r = 1/ClickArrival$ (respectively $1/WordArrival$). (Mean: $1/r$ and Variance: $1/(r^2)$). The density function of the inter-arrival time between 2 events is $f(x) = r * \exp(-r * x)$, if x is greater than 0 and $f(x) = 0$ elsewhere. The inter-arrival time follows an exponential law, for the words and also for the clicks. If the proximity time between a word event and a click event is below ProxyTime and if these two events verify the grammatical and semantic conditions then these two events are fused into one command. The transitions drawn with bold dashed lines model the PCA components distributed over the network. Transitions, with bold lines, model the speech LA components in Figures 12 (a) and (b). The mouse click LA is reduced to a simple thread: 'ThreadofClicks'. The figures 11 (a), (b) and (c) show the simulation results for WordArrival=ClickArrival=5000ms and ProxyTime=10000ms. Figure 11 (c) presents the number of achieved fusions in the time (or the number of marks in the place 'FusionedMedia' of the CPN). In the same way, a command can be cancelled if the user says the word 'cancel' just after an achieved command (the proximity time between the two events: the command and the word 'cancel' is chosen below (ProxyTime/25)). Figure 11 (b) shows the accumulation of words in the corresponding thread (or the number of marks in the place 'ThreadofWords'). Figure 11 (a) shows the resulting cancelled commands in the time (or the number of marks arrived in the place 'CanceledCommand'). The transition 'RecognitionSystem' (Figure 12 (a)) assigns a random label 'wtype' to each word present in the place 'WaitRecognition'. This random assignation does not model a real flowing speech because automatic modeling of user speech is outside the scope of this paper.

The symbol \boxed{HS} in LA transition means that such transition is a Hierarchical Substitution ones. The network in Figure 12 (b) describes interactions at a sublevel of the network in Figure 12 (a).

The symbol \boxed{FG} in identical places indicates that the places are 'global fusion places' [7]. These identical places are simply a unique resource shared over the net by a simple graphical artifact: the representation of the place and its elements is replicated with the symbol \boxed{FG} .

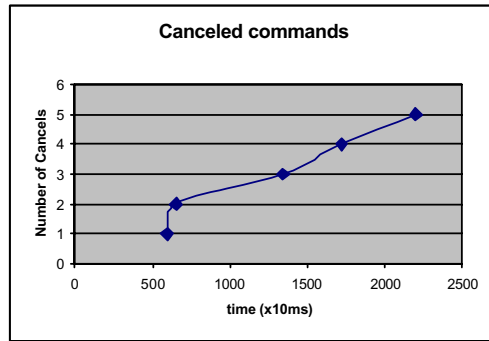


Fig. 11. (a) Simulation Results: Canceled commands

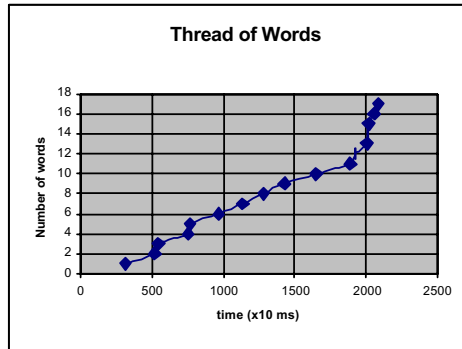


Fig. 11. (b) Simulation Results: Thread of words.

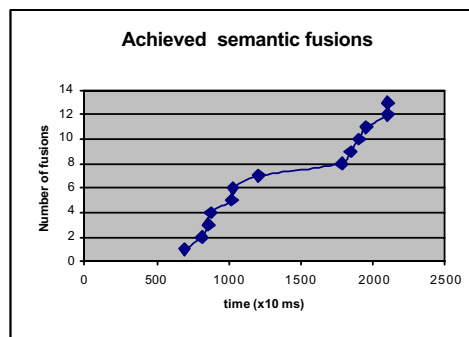


Fig. 11. (c) Simulation Results: Achieved fusions.

Modeling with timed colored Petri nets leads multiple advantages. First of all, These nets can validate a model of timed fusion engine (as shown by the results in figures 11 (a), (b) and (c)). The arrival time between two consecutive events and the processing time by an agent are adjustable. These settings can follow laws of probability (exponential, Erlang etc.). Secondly, the nets are suitable to a distributed modeling where each transition assumes the function of a specialized agent. The function of each agent is easily modifiable (by changing the guard

conditions or the code associated to the transition). Besides, modeling a 'fission' (opposite process of a fusion) is simple to implement because the colorsets of fused marks can keep all the information for that purpose. For example, on Figures 12 (a) and 12 (b), the input arc inscription (to the '**FusionedMed**' place) has event numbers **m** and **n** corresponding to pronounced word and click events respectively. Finally processing concurrent independent tasks is easy.

6 Conclusion

In this paper we have proposed a new agent based architectural paradigms for multimedia multimodal fusion purpose. These paradigms lead to new generic structures that unify the several applications in multimedia multimodal dialog. They also offer to developers a framework specifying different functionalities used in multimodal software implementation. In a first phase, we have gathered the main common requirements and constraints that multimodal dialogs need. We have then identified two interaction types related to the early and late fusions. After identifying the generic recurrent characteristics shared by all modalities, each input media is associated to a specific Language Agent (LA). The LAs are interconnected directly or through a Parallel Control Agent (PCA) to perform the dialog. The architecture of the PCA is decomposed into intelligent components, which provides a modular structure. These components manage temporal, redundant and grammatical conflicts. The proposed architectures are modeled with timed Colored Petri Networks and support both parallel and serial fusions. A typical simulation example, with random input flows, has illustrated our approach. The simulation allowed us to verify that the proposed architectures performed correctly the expected fusions.

Acknowledgments : We wish to acknowledge the financial support of the Natural Sciences and Engineering Research Council (NSERC) of Canada

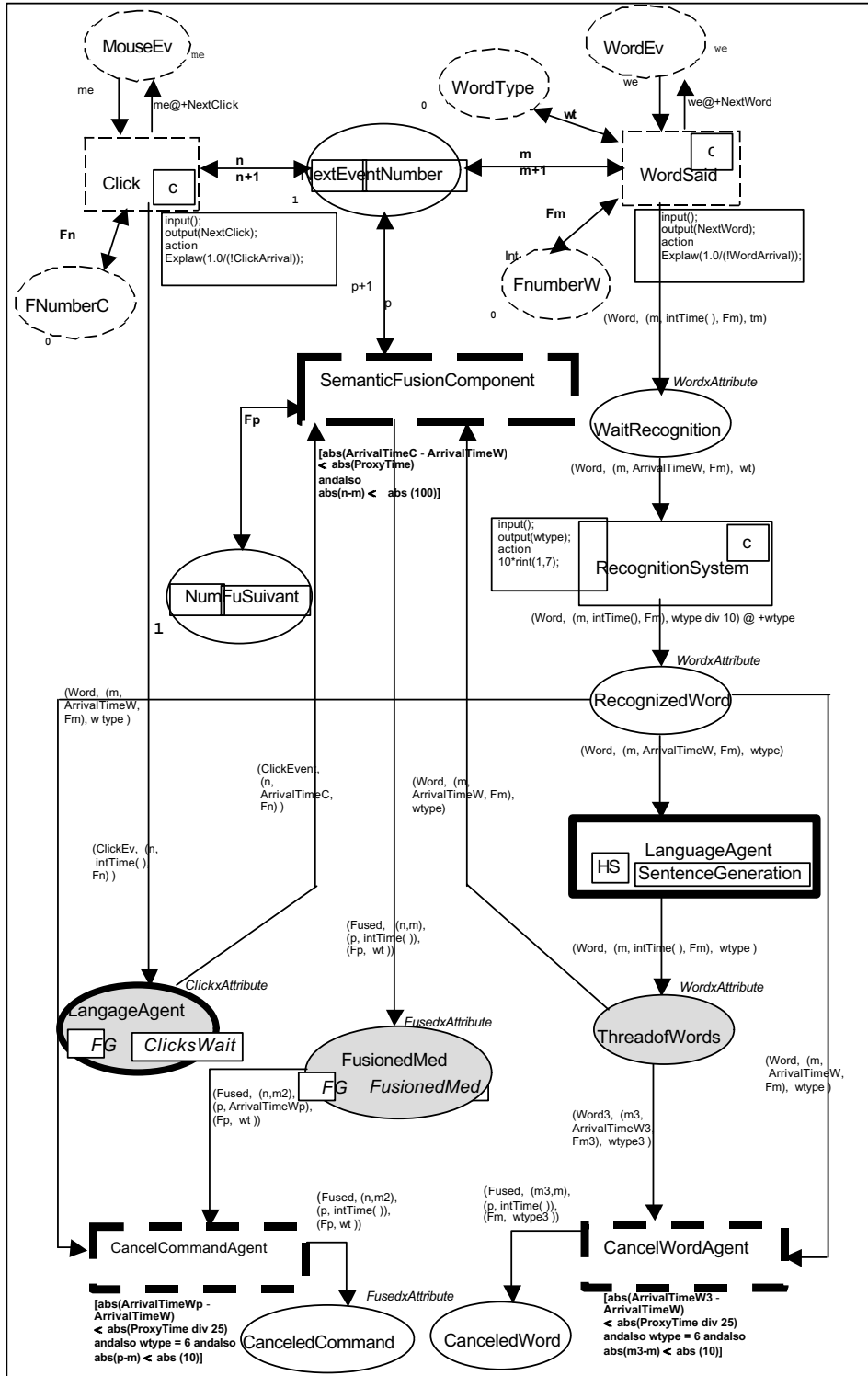


Fig. 12. (a). Bimodal fusion dialog.

REFERENCES

1. Bolt, R.A., *Put that there: Voice and gesture at the graphics interface*, ACM Computer Graphics 14,3, 262-270, 1980.
2. Crowley, J.L. and Bérard, F. *Multimodal tracking of faces for video communications*, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97), San Juan, IEEE Press, NY, June, 1997.
3. Bellik Y., Burger D., *Multimodal Interfaces: New Solutions to the Problem of the Computer Accessibility for the Blind*. Proc. CHI'94, Boston, 24-28 April 1994.
4. McGee, D.R., Cohen, P.R., and Wu, L., *Something from nothing: Augmenting a paper-based work practice with multimodal interaction*, in the Proceedings of the Conference on Designing Augmented Reality Environments, ACM Press, Helsingor, Denmark, 71-80, April 12-14, 2000.
5. Jensen, K., *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Volume 1, Basic Concepts*. Monographs in Theoretical Computer Science, Springer-Verlag, 2nd corrected printing, ISBN: 3-540-60943-1, 1997.
6. Jensen, K., *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Volume 2, Analysis Methods*. Monographs in Theoretical Computer Science, Springer-Verlag, 2nd corrected printing, ISBN: 3-540-58276-2, 1997.
7. Jensen, K., Christensen, S., Huber, P and Holla, M., *Design/CPN Reference Manual*, Department of Computer Science, University of Aarhus, Denmark, <http://www.daimi.au.dk/designCPN/>, 1995.
8. Oviatt, S.L., *Multimodal Signal Processing in Naturalistic Noisy Environments*, In B. Yuan, T. Huang and X. Tang Eds., Proceedings of the International Conference on Spoken Language Processing (ICSLP'2000), Vol. 2, pp. 696-699, Beijing, China: Chinese Friendship Publishers, 2000.
9. Oviatt, S.L., *Multimodal System Processing in Mobile environments*, Proceedings of the Thirteenth Annual ACM Symposium on User Interface Software Technology UIST'2000, pp. 21-30, New York: ACM Press, 2000.
10. Hutchins, E. L., Holland, J. D. and Norman, D. A., *Direct manipulation interfaces*, In Norman, D. A. and Draper, S. W. Eds., User centred system design: new perspectives on human computer design, Hillsdale, NJ, Lawrence Erlbaum, 1986.
11. Oviatt, S.L., Cohen, P.R., Wu, L., Vergo, J., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J. and Ferro, D., *Designing the user interface for multimodal speech and gesture applications: State-of-the-art systems and research directions*, Human Computer Interaction, vol. 15, no. 4, pp. 263-322, 2000.
12. Bregler, C., Manke, S., Hild, H., and Waibel, A. *Improving connected letter recognition by lip reading*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, 1, pp. 557-560. IEEE Press, 1993.
13. *Projet AMIBE*, Rapport d'activité, GDR |n° 9, GDR-PRC Communication Homme-Machine, CNRS, MESR, 1994, pp. 59-70 (Technical Report in french), 1994.
14. Oviatt, S. L., *Mutual disambiguation of recognition errors in a multimodal architecture*, Proceedings of Conference on Human Factors in Computing Systems: CHI '99, New York, N.Y., ACM Press, 576-583, 1999.
15. Coutaz, J., Nigay, L., *Les propriétés CARE dans les interfaces multimodales*, IHM'94. Sixièmes journées sur l'ingénierie des Interfaces Homme-Machine, Lille, 89 Déc, (French paper), 1994.
16. Jennings, N. R. and Wooldridge, M. J., "Applications of Intelligent Agents" in "Agent Technologies: Foundations, Applications and Markets", Eds. N. R. Jennings and M. Wooldridge, 3-28, 1998.
17. Weiss, G., *Multiagent Systems*, MIT-Press Ed., 1999.

18. Bird, S.D., *Toward taxonomy of multi-agents systems*, International Journal of Man-Machine Studies, 39, 689-704. 1993.
19. Bond, A.H. and Gasser, L., *Readings in Distributed Artificial Intelligence*, San Mateo, Calif.: Morgan Kaufmann, 1988.
20. Ishida, T., *Real-Time Search for Learning Autonomous Agents*, Kluwer Academic Publishers, 1997.
21. Muller, H. J., *Negotiation principles*, In G. M. P. O'Hare and N. R. Jennings, eds, Foundations of Distributed Artificial Intelligence, pp. 211-229, Wiley, 1996.
22. Cohen, P. R., Levesque, H. R., and Smith, I., *On team formation*, Hintikka, J. and Tuomela, R. (Eds.) Contemporary Action Theory. Synthesis, 1997.
23. Tambe M., Johnson W. L., Jones E. M., Koss F., Laid J. E., Rosenblum P. S., and Schwamb K., *Intelligent agents for interactive simulation environments*, AI Magazine, 16(1), pp. 15-39, 1995.
24. Abrial J.-R, *The B-Book: Assigning Programs to Meanings*, Cambridge University Press, 1996.
25. Bellik Y., : Thèse de Doctorat de l'Université de Paris XI, spécialité informatique.« *Interfaces multimodales : concepts modèles architectures* » soutenue le 30Mai 1995 par Yacine Bellik PHD Thesis of univerity au ParisXI (France)
26. Chen, S.-C. and Kashyap, R. L., *Temporal and spatial semantic for multimedia presentations*, International Symposium on Multimedia Information Processing, pp. 441-446, Dec.11-13, 1997.