Vol. 4, No. 5 May 2013 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

Multimodal Fusion for Interaction Systems

[1]Ahmad Wehbi, [2] Atef Zaguia, [3]Amar Ramdane-Cherif, [4] Chakib Tadj
[1,2,4] MMS Research Group, Université du Québec, École de technologie supérieure, Montréal, Canada
[1,3] LISV Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines, France,

[1] ahmad.wehbi.1@ens.etsmtl.ca, [1] ahmad.wehbi@lisv.uvsq.fr, [2] atef.zaguia.1@etsmtl.ca, [3] rca@lisv.uvsq.fr, [4] ctadj@ele.etsmtl.ca

ABSTRACT

Researchers in computer science and computer engineering devote now a significant part of their efforts in communication and interaction between human and machine. Indeed, with the advent of real-time multimodal and multimedia processing, computer is no longer seen only as a calculation tool, but as a machine of communication processing, a machine that accompanies, assists or promotes many activities in daily life. A multimodal interface allows a more flexible and natural interaction between a user and a computing system. It extends the capabilities of this system to better match the natural communication means of human beings. In such interactive system, fusion engines are the fundamental components that interpret input events whose meaning can vary according to a given context. Fusion of events from various communication sources, such as speech, pen, text, gesture, etc. allow the richness of human-machine interaction

This research will allow a better understanding of the multimodal fusion and interaction, by the construction of a fusion engine using technologies of semantic web domain. The aim is to develop an expert fusion system for multimodal human-machine interaction that will lead to design a monitoring tool for normal persons, seniors and handicaps to ensure their support, at home or outside.

Keywords: *Fusion, multimodal interaction, ontology, modalities, OWL API, fusion models*

1. INTRODUCTION

The multimodal interaction systems allow users to interact with computers using different input modalities (speech, gesture, vision, etc.) And output channels (text, graphics, sounds, avatars, synthesized speech, etc.). This type of user interface is not only beneficial to improve accessibility, but also for convenience (the natural recognition of an input mode) and flexibility (adaptation to the context and the environment). One of the objectives of the research HMI (human / machine interface) is to increase communication skills between man and computer. This involves developing specifications, tools, software and methodologies for the coordination of multiple media and methods of communication, such as speech, gestures, etc. The user has the choice of modality that may prefer to use at any given time. This is called equivalence, in the sense that the terms are not equal, but allow the user to get the same result from the system. It is therefore important when designing the system to conduct a needs assessment and analysis of the task, in order to offer a combination of media and appropriate modalities.

This paper proposes a new solution that facilitates work of a fusion engine, using an ontology that describes the environment and takes into account the context and well-defined pre-conditions. It is based on semantic web languages based on W3C standards. This makes the proposed solution applicable to many systems (that use these standards) and prevents it from being limited to a specific domain of application. This ontology will include knowledge and conditions that describe different contexts and fusion rules needed for fusion.

The paper is organized as follows. In section 2, we will review some previous architectural designs and show their weaknesses. In section 3, we will present the proposed approach and describe the environment and the fusion engine in detail. Section 4, a study case will be presented. In Section 5, the implemented prototype will be presented and finally, in section 6, we will end with the conclusion.

2. RELATED WORK

The multimodal concept allows the user to sending commands to a machine as well as the machine sending output to the user using modalities. Such modalities could be a touch screen, stylus and man's natural modalities, such as speech, eye gaze and gestures. In recent times, multimodal fusion has gained much attention of many researchers due to the benefit it provides for various multimedia analysis tasks. Multimodality allows the invocation of other means when some other modalities are not available or possible to use. Several multimodal systems have been proposed after Bolt's pioneering system [1]. Speech and lip movements have been merged using histogram techniques [2], multivariate Gaussians [2], artificial neural networks (ANNs) [3], [4] or hidden Markov models (HMMs) [2]. In all these systems, the modalities' probabilistic outputs have been combined assuming conditional independence by using either the Bayes' rule or a weighted linear combination over the mode probabilities for which the weights were adaptively determined.

Various multimodal applications [5], [6] are conceived and are effective solutions for users who have constraints such as the impossibility of using a keyboard or a mouse [7], having visual handicap [8], being in move, using mobile devices [9], and being weak or disabled [10].

QuickSet [11] is a multimodal pen-gestures and spoken input system for map-based applications. A multi-dimensional chart parser semantically combines the statistically ranked set of input representations using a declarative unification-based grammar [12]. The EMBASSI system [13] combines speech, pointing gesture and the input from a graphical GUI into a pipelined architecture. The Smartkom [14] is multimodal dialogue system that merges gesture, speech and facial expressions for both input and output via an anthropomorphic and affective user interface. In both systems, input signals are assigned a confidence score that is used by the fusion module to generate a list of interpretations ranked according to the combined score. Barboni et al [15] proposed an ontological solution for a system called SmartWeb. This system is based on question answering technology that combines different kinds of domain ontologies into an integrated and modular knowledge base. The main problem of this approach is the specification of its design, which makes the proposed solution very limited. The architecture of HephaisTK system developed by Dumas et al [16] is based on agents that manage individual modality recognizers, receive and encapsulate data from the recognizers, and send them to an individual central agent named the "postman". However, this architecture needs a configuration file to be specified for describing the human-machine multimodal dialog desired for the client application, and for the specification to which recognizers need to be used.

In [17], two statistical integration techniques have been presented: an estimate and a learning approach. The estimate approach makes use of a multimodal associative map to express, for each multimodal command, the meaningful relations that exist between the set of the single constituents.

During multimodal recognition, the posterior probabilities are linearly combined with mode-conditional recognition probabilities that can be calculated from the associative map. Mode-conditional recognition probabilities are used as an approximation of the mode-conditional input feature densities. In the learning approach, called Members to Teams to Committee (MTC), multiple teams are built to reduce fusion uncertainty. Teams are trained to coordinate and weight the output from the different recognizers while their outputs are passed on to a committee that establishes the N-best ranking.

Multimodal interface tools are currently few in numbers and they address to a very specific technical problem such as media synchronization [18], or they are dedicated to very specific modalities. For example, the Artkit toolkit which is designed to support direct manipulation augmented with gesture only [19] or MATIS (Multimodal Airline Travel Information System) which allows a user to retrieve information about flight schedules using speech, direct manipulation, keyboard and mouse, or a combination of these techniques [20].

Having these weaknesses taken into account, we come up with a proposed architecture that addresses these issues. This is done through environment description and understanding that contains an increasing number of modalities, events and fusion models. The adoption of our multimodal fusion system will be a new approach that facilitates the work of a fusion engine by giving it the most meaningful combinations of events.

3. PROPOSED APPROACH

In this section, we will describe our proposed architectural design as a solution to the described problems of previous architectures. The proposed solution has three main characteristics:

- **Openness**: handling a large number of modalities which prevents the restriction in its application to specific domains.
- **Flexibility**: using ontology makes the description of an environment and its scenarios easier.
- **Consistency**: by the description of the most potential objects and scenarios of the environment.

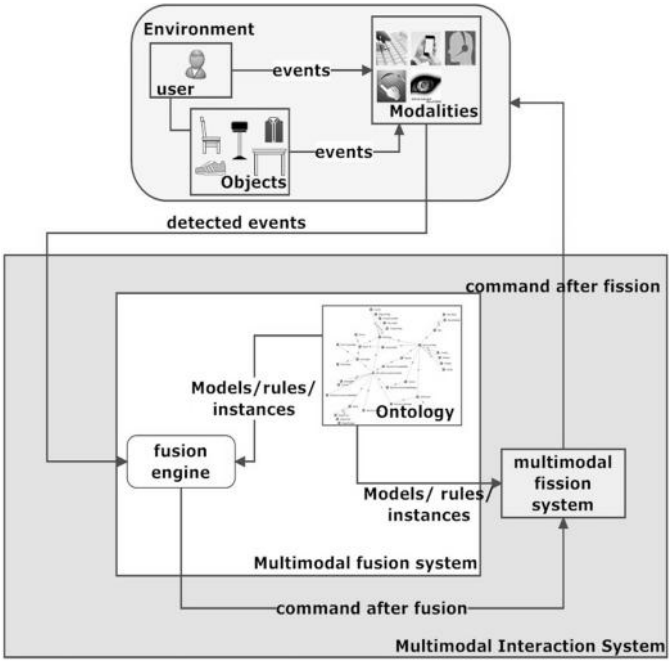

Fig 1: General approach of multimodal fusion system

A general overview of the architectural design is shown in

Fig 1 . It illustrates the different components of the environment and show how a user or objects supplies events to the fusion engine using different modalities leading to the fusion process inside the multimodal fusion system. These components are as follows:

- **Environment**: it could be indoor or outdoor and contains a user, objects and modalities.

- **Multimodal Interaction System**: it contains the Multimodal fusion and fission systems

- **Multimodal fission system**: it is the part responsible of executing the commands after understanding them in the fusion process.

- **Multimodal fusion system**: it is the part responsible of understanding the environment and the user commands. It contains the fusion engine, the ontology

The multimodal fusion system will merge the events coming from the environment to produce a meaningful command. The multimodal fission system will interpret this command and divide it into elementary tasks in order to be executed in the environment using output modalities.

In this paper, we will not deal with the fission system; we are only interested in the fusion system that merges events coming from input modalities.

3.1 Ontology

The role of the ontology is to describe semantically the environment surrounding the user. It is described using a tool called PROTÉGÉ [21]. This tool is based on Ontology web language (OWL) [22].

3.1.1 Concepts of the Ontology

The main concepts that describe the environment are represented in the **Fig** 2; it's composed of:

- **Coordinates:** used to locate different locations of objects, places and persons inside the environment.

- **Context:** it describes parameters that must be taken into account when detecting events from input modalities.

- **Modality:** it contains the different input modalities used by a user for sending events (see
- **Fig** 4**)**.

- **Event:** it describes a set of events that can be sent by a user using modalities and also by the environment itself.

- **Model:** it contains a set of models representing different combinations of events that forms meaningful commands.

- **Time:** it represents the time constraints for commands and their events.

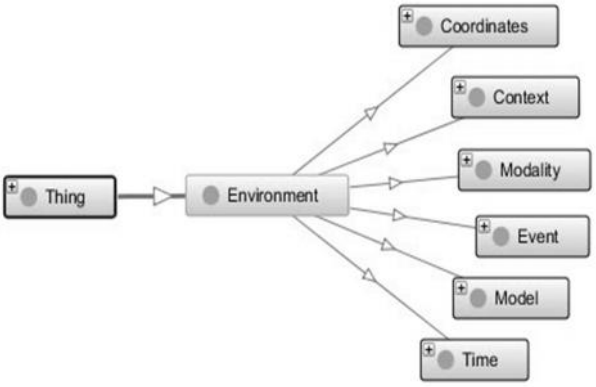

Fig 2: The main concepts of the ontology

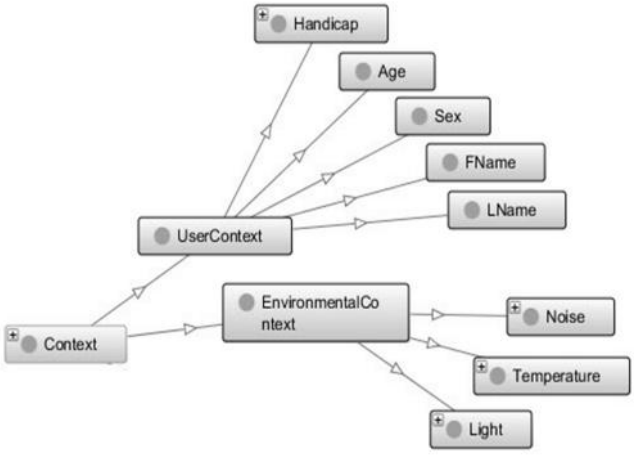

Fig 3: The concept Context

Fig 3 shows the different subclasses of the class Context, it's formed by two subclasses:

- **User Context:** it describes the user profile, especially if he has a handicap or no, because this will affect the choice of modality

- **Environmental Context:** it describes the lightness, darkness, noise and temperature level, etc. inside an environment.
-

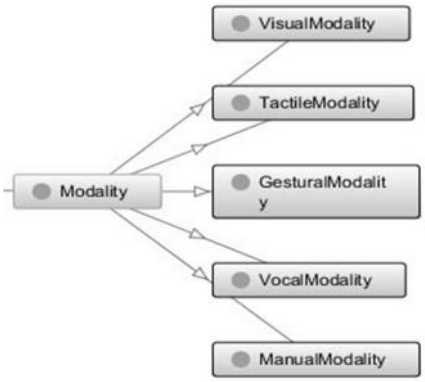

Fig 4: the Concept Modality

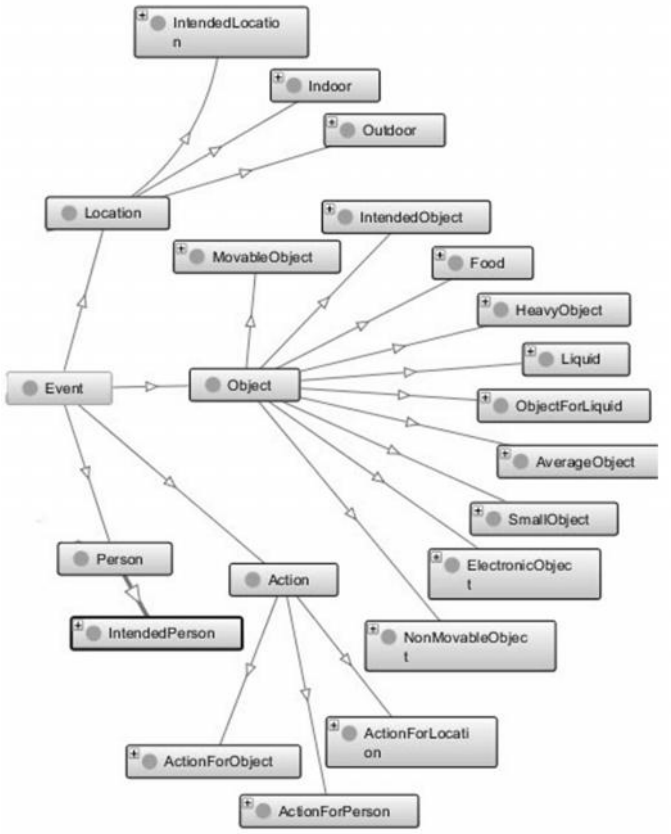

Fig 5: The concept Event

Fig 5 represent the different subclasses of the concept Event. 4 subclasses are defined:

- **Location**: represents the different places mentioned inside a user command. It contains 3 subclasses, Outdoor, Indoor and Intended Location which is used when pointing to a location.

- **Object:** represents the different objects mentioned inside a command by the user. Objects are divided by size and type. it contains NonMovableObject, Movable Object, ObjectForLiquid, Liquid, Food, Small Object, Average Object, Heavy Object, Electronic Object and Intended Object. The intended Object class is used when pointing to an object instead of naming it.

- **Action:** represents the order mentioned inside the user command. It contains Action For Person, Action For Location and Action For Object which contain the two subclasses Action For Movable Object and Action For Non Movable Object. These classes represent the actions related to a person, a location and an object.

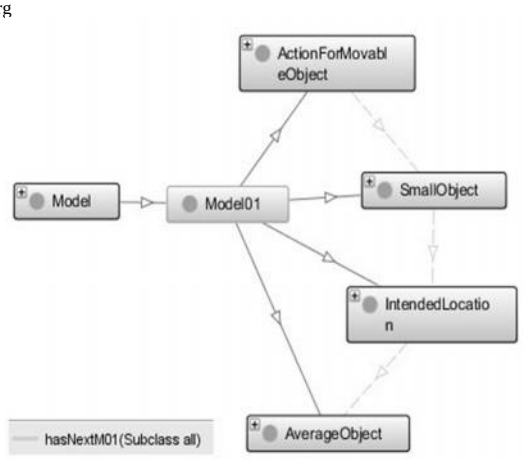

Fig 6: The concept Model01

In Model class, thirty models were created (Model01, Model02… Model30). They represent the different combinations of events that form meaningful commands. An example of models is shown in

Fig 6, Model01 represent a combination of Action For Movable Object that has Next Small Object that has Next Intended Location that has Next Average Object ex: the command "put the cup on the table", put is an action (Action For Movable Object) that has next a cup (movable small object) cup has next an on (intended location)and on has next a table) (average object).

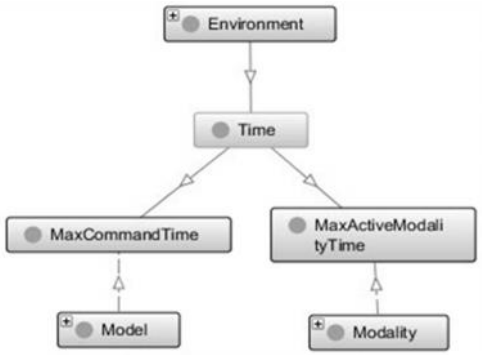

Fig 7: The concept Time

Fig 7 represent the Time class and its subclasses, Max Commad Time is the maximum time allowed for a complete command, for example the maximum time needed to say give me that cup is 10s. Max Active Modality Time is the maximum time allowed so that another modality intervenes and be active. Every model has a Maximum command time and every modality has a maximum active modality time. The time is an important parameter that must be taken into account, because it helps the system to identify if an event belongs to a command or no.

3.1.2 Instances of the Ontology

Vocabulary in our modeling is the explicit representation of knowledge that designates an object, a person, a place, their coordinates, an action, and a modality inside the

environment. Vocabulary is defined simply to say that a key, a book, a watch for example are part of the class Small Object. Vocabulary defined in the ontology as instances. An example of instances is shown in the **Table** 1, **Fig** 8 and **Fig** 9 below.

Table 1: Sample of instances

Class	Instances
Action For Movable Obeject	put, drag, take, bring, etc.
Action For Non Movable Obj	clean, close, open, etc.
Action For Location	locate, search, find, etc.
Action For Person	call, ask, contact, etc.
Average Object	box, chair, table, desk, etc.
Small Object	cup, pen, mobile, paper, etc.
Heavy Object	door, wall, refrigerator, etc.
NonMovableObject	wall, door, window, etc.
Movable Object	bottle, fork, cup, etc.
Liquid	water, milk, jus, wine, etc.
Food	banana, meat, chicken, orange, etc.
Electronic Object	microwave, television, radio, etc.
Intended Object	that, it, this.
Indoor	Kichten, bedroom, bathroom, etc.
Outdoor	road, backyard, etc.
Intended Location	there, here, under, on, after, behind, etc.
Person	brother, father, mother, son, etc.
Intended Person	us, me, him, her, etc.

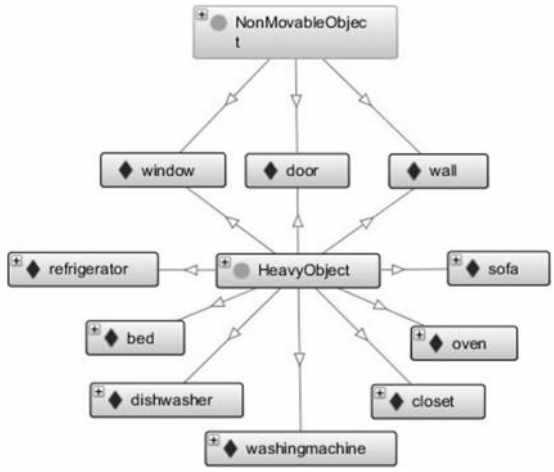

Fig 8: Vocabulary of Heavy Object concept

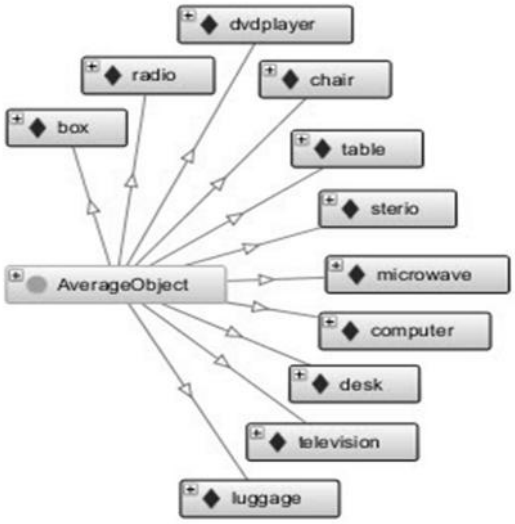

Fig 9: Vocabulary of Average Object concept

3.1.3 Properties of the Ontology

After the creation of concepts and instances, in this section we present the semantic relationships between the classes themselves, the classes and instances and the instances themselves. Two types of relationships have been used in the ontology, properties of objects (between classes and / or instances) and data properties (between instances and their values). Semantic relations are very important because they take the role of knowledge mapping, they can situate knowledge with respect to other. An example of properties is shown in Fig 10 below.

Fig 10 shows objects and data properties between instances. The class Vocal Modality has an instance Voice_Sensor_Living_Room; it means that a vocal sensor is located in the living room. This instance has an object property with the instance Noise From Outside of the class Noise. Similarly Voice_Sensor_Living_Room has another object property has User Context with both deaf and mute instances of the class Handicap. Noise From Outside has a data property has Noise Level which has a value equal to 5. Deaf and mute have another data property has Handicap as false in both cases. These semantic relationships are used by reasoners in order to understand the environment, the goal is to determine the type and level of noise detected (type: Noise From Outside, Level: 5) and the type of handicap for a user. The voice mode is affected by two types of handicaps deaf and mute. A deaf or mute person cannot send voice commands to the system. If both instances (deaf and mute) have a false value, and if the noise level detected less than 5 for example, the vocal modality well be activated, otherwise it will be disabled. Note that the values of different contexts can be changed according to user's profile.

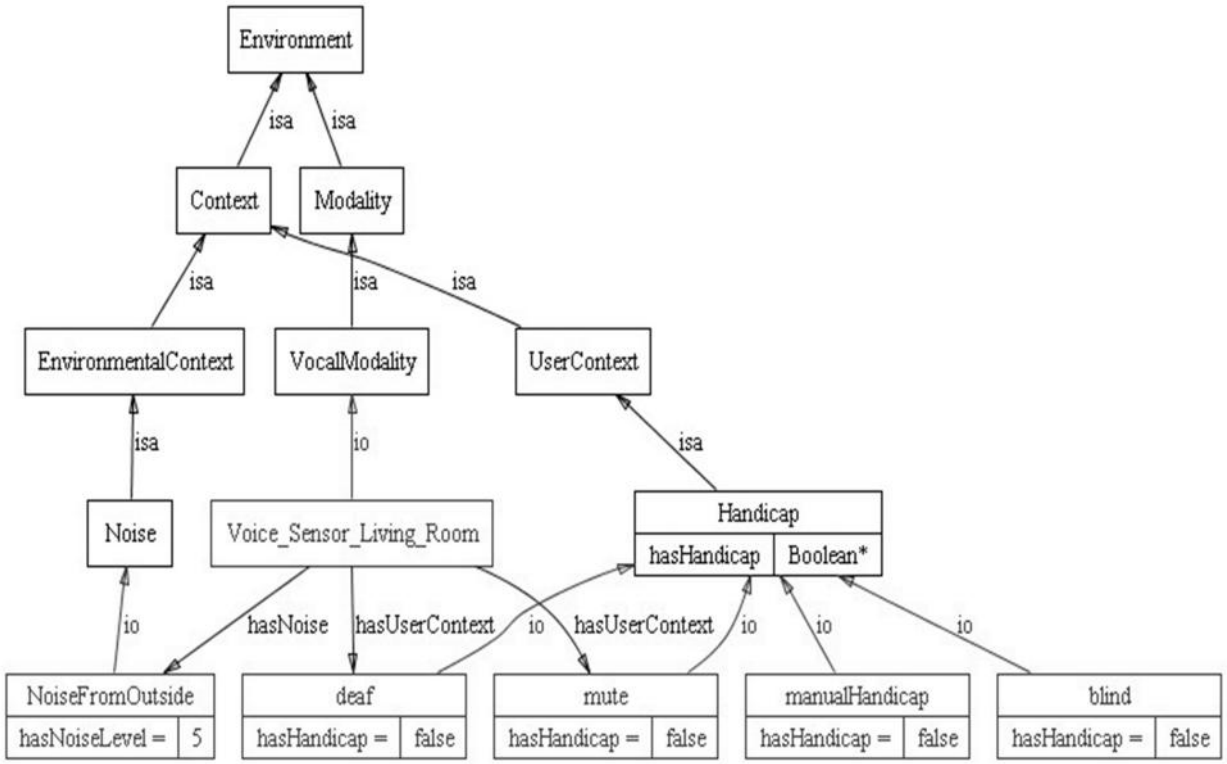

Fig 10: Properties that relate the vocal modality with contexts

Fig 11 shows an example of relations between instances of the class Model02. Model02 is formed by three subclasses Action For Movable Object, Average Object and Intended Location; they have as instances, get, box and here respectively. HasNextM02 is a relation of type object property that has been created to define the order of the instances. This means that get is followed by box and box is followed by here. This kind of relations helps to better understand the sequence of events in a specific command.

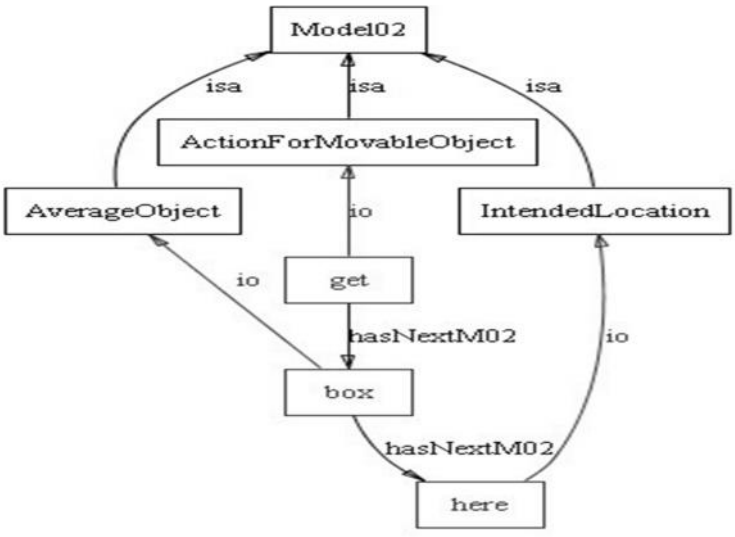

Fig 11: relation between instances of the class Model02

3.2 Fusion Engine

Fusion engine is a module responsible for the merging of different events coming from different modalities. Two modalities are needed at least so that the fusion can be occurred, otherwise, there is no need to fusion because it will be a mono-modal and not a multimodal case.

A set of preconditions and constraints must be taken into account to do multimodal fusion:

- **Vocabulary Checking**: to verify if an event exist in our defined vocabulary or not. This is made it by comparing the event with the instances of the ontology. If the event is founded, then the next precondition is tested, else the fusion mechanism cannot be achieved. For example if a vocal modality sends the word take said by a user, the system check if this word exist as instance of the Action class, if yes, the system continue the verification of other precondition and if no, the system reject it.

- **Events order**: to verify if events sent from modalities are respecting the expected order defined in different models of the class Model in the ontology. It is sufficient to declare the SQWRL queries in order to select an appropriate model to a combination of events detected by modalities. Each model has its own query. Thirty query were created for each model

- **Semantic checking:** to verify if events are semantically correct between them. It's done by verifying different constraints defined in the ontology as Object Properties and Data Properties (relations between classes or between instances, ex has Next, has Modality, has Time, etc.) and as SWRL rules. This verification can be done using the PELLET inference engine [23] of PROTÉGÉ by checking the consistency, the taxonomy and the inferences between different classes of the ontology. SWRL rules are tested using the jess engine [24] which is a tool that have a plug-in in POTEGE. The semantic verification is very important otherwise the system will consider any command as a correct command. For example, cup and paper are two instances of the class Small Object, but without the semantic checking, a command like "give him a paper of water" will be correct, because it match a model declared in our ontology

  (ActionForMovableObject → Person → MovableObject → MovableObject) while the correct command is "give him a cup of water". That's why, a set of SWRL rules are defined to check these cases, and verify that for example, before any kind of liquid, there must be a cup of a bottle, otherwise the command will be rejected by the system. The SWRL rule needed for this example is:

$$\text{Liquid}(?y) \wedge \text{Small Object}(?x) \wedge \text{hasObject}(?x, ?y) \rightarrow \text{hasNext30}(?x, ?y) \quad (7)$$

  The result of the execution of this rule shows that a cup or a bottle is followed by a type of liquid using the relation haLiquid (example: cup has Next soda, bottle has Next jus).

- **Temporal conditions:** to verify the maximum time allowed between two active modalities and the time of a command itself. Here we shall illustrate various variations of time frame which will decide if fusion is possible or not. For all the cases involved, the following notations shall be used: M_i = modality i, t_1M_i = arrival time of modality i, t_2M_i = end time of modality i, $t_{maxActiveModalityTime}$ = maximum time delay allowed for another modality to be involved.

Fig 12 shows some of the possible cases:

- Case A: Modality 1 arrives at t_1M_1 and ends at t_2M_1. Another modality 2 arrives at t_1M_2 and ends at t_2M_2. They, however, ΔT is greater than the Max Active Modality Time, so each modality is uni modal and it is treated separately. Hence,

$$f_1: C_1 = M_1 \text{ and } f_2: C_2 = M_2$$

- Case B: Modality 1 and modality 2 arrive separately but ΔT is less than Max Active Modality Time. In this case, a test is needed to determine if fusion is possible. In the affirmative case, the resulting function would be:

$$f: C = M_1 + M_2$$

- Case C: Modalities 1 and 2 arrive within the same time frame. Here, fusion is obvious and the resulting function is $f: C = M_1 + M_2$.

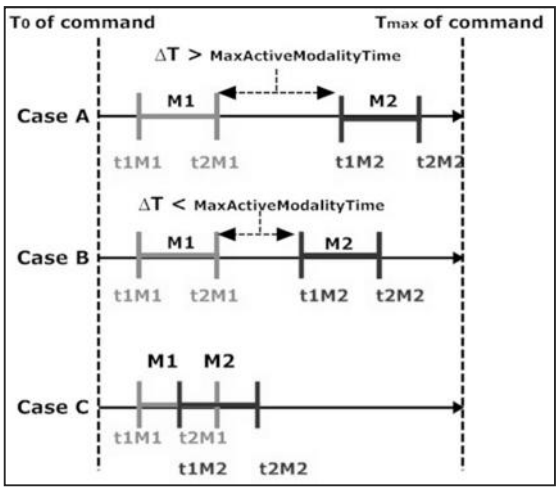

Fig 12: Temporal aspect of events

Once we have specified the preconditions needed, an algorithm is defined to checks these preconditions inside the ontology. (

Table 2). This algorithm describes the different steps followed by the system to verify if each condition is respected or not. If one of these condition is not respected, the fusion will not occur, else if all of them are true the fusion occur and the system understand the user command.

Table 2: fusion algorithm

	Fusion Algorithm
1	Declaring Command as string;
2	String instance=words of the command;
3	For (each instance) {
4	Get Instance Of (Instance [i]); // get the instances of the command from the ontology
5	//check if events of the commands exist as instances of the ontology
6	If (instance [i] = instances in the ontology) {
7	Memorize instance [i];
8	Check instance [i+1]; }
9	//check maximal command time;
10	Get Start Time of event1;
11	Get End Time of last event;
12	If (end Time – start Time) <=Max Command Time){
13	//check time between modalities
14	ΔT= start Time of Mi+1 – end Time of Mi;
15	If (ΔT < Max Active Modality Time){
16	//Match a model of ontology with the command
17	Get Classes Of Instance;
18	// check their order if exists in a model defined inside the ontology
19	If (Matching ok);
20	Result of fusion;}}
21	Else
22	No fusion;
23	Else
24	No fusion;
25	Else
26	No fusion;
27	Else
28	No fusion;}

4. CASE STUDY

In this section, we present the scenario "get that here" to show how to apply the fusion algorithm. To do this, we must define our events and contexts. We supposed that the type of light where the system exists is a living room light and its level is equal to 6, a noise is coming from the TV in the room with a level 3. The user does not have any type of handicaps. So we deduce that the system is in a well lighted and quiet environment and the user has no handicap.

Executing the query below shows that, no modalities affected by the light should be disabled because the light level is greater than 5 (5 is defined during the configuration of the system during the first use as a good level of lightness).

Modality (?m) ∧ Light(Light_Living_Room) ∧ has Light(?m, Light_Living_Room) ∧ has Lightness Level(Light_Living_Room, 6) ∧ swrlb:lessThan(6, 5) → sqwrl:selectDistinct(?m)

Executing the query below shows that, no modalities affected by noise should be disabled because the noise level is less than 6 (6 is defined in the configuration of the system during the first use).

Modality(?m) ∧ Noise(TVNoise_Living_Room) ∧ has Noise(?m, TVNoise_Living_Room) ∧ has Noise Level(TVNoise_Living_Room, 3) ∧ swrlb: greater Than(3, 6) → sqwrl: select Distinct(?m)

Executing the query below shows that, no modality is affected by type of handicaps since they all have a value of 0, because the user is normal and has no handicaps

Modality(?m) ∧ Handicap(?h) ∧ hasUserContext(?m, ?h) ∧ hasHandicap(?h, ?han) ∧ swrlb:equal(?han, 1) → sqwrl:selectDistinct(?m)

The **Table 3** shows the different events detected by a gestural and voice mode

Table 3: detected events

Modality	**Event**	**Start Time**	**End Time**
Vocal Modality	Get	0.1	0.3
Vocal Modality	That	0.4	0.6
GesturalModality	(9,15,8)	0.65	0.69
Vocal Modality	Hello	0.7	0.8
Vocal Modality	Here	0.9	1.1
GesturalModality	(11,20,10)	1.5	2
Vocal Modality	Hi	0.75	0.78
Vocal Modality	Zzzz (bruit d'une porte)	0.95	0.97
Vocal Modality	No	1.5	2.7

The vocabulary is verified by executing the query below for each detected event and recognized events are shown in the

Table 4

Event (?m) ∧ tbox:equalTo(?m, événement) → sqwrl:selectDistinct(?m)

Table 4: recognized events

Event	**Class**
Get	ActionForMovableObject
that	Intended Object
(9,15,8)	Coordinates
Here	IntendedLocation
(11,20,10)	Coordinates

The events hello, hi, zzzz, No are removed because they are not defined as instances in the ontology and the system could not recover their classes.

The fusion engine tries to find the model that defines such an order ActionForMovableObject → Intended Object → IntendedLocation which is as follows:

ActionForMovableObject(?x) ∧ Intended Object(?y) ∧ IntendedLocation(?z) ∧ hasNextM08(?x, ?y) ∧ hasNextM08(?y, ?z) ∧ tbox:isDirectSuperClassOf(?m, ActionForMovableObject) ∧ tbox:isDirectSuperClassOf(?m, Intended Object) ∧ tbox:isDirectSuperClassOf(?m, IntendedLocation) → sqwrl:selectDistinct(?m)

Running this query gives us the model 8 (Model08) from the ontology. So this series of events has been identified.

Finally, it remains to verify the temporal aspect of events. Two operations are necessary:

- The subtraction of the end time of the last event and the start time of the first one to determine if the maximum time of a command is respected and
- Subtracting the time between two consecutive events to know whether the maximum time between events is respected. The maximum time of a command and the maximum time between the events are defined according to the user during the initial configuration of the system. Supposing that the maximum time a command is set to 10 seconds and the maximum time between two events is 3 s.
-

2 - 0.1 = 1.9 s < 10 s: maximum time of a command is respected
0.4 - 0.3 = 0.1s < 3 s: time respected between that and get
0.65 – 0.6 = 0.05s < 3 s : time respected between (9, 15, 8) and that
0.9 – 0.69 = 0.21 s < 3 s : time respected between here et (9, 15, 8)
1.5 – 1.1 = 0.5 s < 3 s : time respected between (11, 20, 10) and here

In conclusion, the temporal aspect is well respected and all other conditions are met. The fusion took place and the fusion engine got the command "get that here", the coordinates of that (9, 15, 8) and here (11, 20, 10).

Now, assuming that, we still have the same contexts and the system detects two events get and me that are defined in the ontology and belong to two classes ActionForMovableObjet and IntendedPerson respectively. Semantics is respected here. There is no contradiction and relationships are consistent. But, when the system selects the model, it will not find the model associated with such order, because in reality, this series of events is not complete. The command get me has no sense. So the engine removes it and the fusion engine will not check the other conditions. Thus, the fusion did not occur.

5. IMPLEMENTATION

In this section, a prototype has been implemented to apply the scenario "get that here" described in the section 4. The goal was to demonstrate that the proposed architecture is applicable in a real system. This prototype focused on the fusion engine, but it can be developed in a comprehensive manner to ensure the different services according to specifications.

The developed system is dedicated to multimodal fusion. The initial objective was to understand the environment and its events.

With OWL API (http://owlapi.sourceforge.net/), the connection between JAVA and the ontology is established. Reasoning and inference of classes and properties have been verified and queries responsible for the selection of modalities and models have been performed.

The programming language used for the prototype is Java (java.com), in the NetBeans platform 6.9.1 (netbeans.org/) it is an integrated development environment (IDE), and offered as open source by Sun. The OWL API (Application Programming Interface OWL) was used to establish the connection between the code and ontology. The OWL API is a Java API for creating, manipulating and serializing OWL ontologies.
Events from environemnt are sent to the system as XML files, there is a parser that transfer these data to the application. The
Table 5 and **Error! Reference source not found.** shows the different XMLfiles used bye the system

Table 5: Events in XML file

```
<events>
  <Comand>
  <Modality>
        <type>VocalModality</type>
        <event>get</event>
        <startTime>0.1</startTime>
        <endTime>0.3</endTime>
    </Modality>
 <Modality>
        <type>VocalModality</type>
        <event>that</event>
        <startTime>0.4</startTime>
        <endTime>0.6</endTime>
    </Modality>
    <Modality>
        <type>GestureModality</type>
        <event>(9,15,8)</event>
        <startTime>0.5</startTime>
        <endTime>0.6</endTime>
    </Modality>
 <Modality>
```

```
        <type>VocalModality</type>
        <event>hello</event>
        <startTime>0.7</startTime>
        <endTime>0.8</endTime>
    </Modality>
 <Modality>
        <type>VocalModality</type>
        <event>here</event>
        <startTime>0.9</startTime>
        <endTime>0.11</endTime>
    </Modality>
        <Modality>
        <type>GestureModality</type>
        <event>(11,20,10)</event>
        <startTime>0.15</startTime>
        <endTime>0.22</endTime>
    </Modality>
    <Modality>
        <type>VocalModality</type>
        <event>hi</event>
        <startTime>0.75</startTime>
        <endTime>0.78</endTime>
    </Modality>
    <Modality>
        <type>VocalModality</type>
        <event>zzz</event>
        <startTime>0.95</startTime>
        <endTime>0.97</endTime>
    </Modality>
    <Modality>
        <type>VocalModality</type>
        <event>no</event>
        <startTime>1.5</startTime>
        <endTime>2.7</endTime>
    </Modality>
  </Comand>
</events>
```

Table 6: Contexts XML File

```
<events>
  <Command>
        <GModality>

<type>Gestures_Modality_Living_Room</type>
              <LightLevel>6</LightLevel>
        </GModality>
        <VModality>
        <type>Gestures_Modality_Living_Room</type
        >
              <NoiseLevel>3</NoiseLevel>
        </VModality>
 </Command>
<contexts>
        <UserContext>
              <blind>false</blind>

<manualHandicap>false</manualHandicap>
              <deaf>false</deaf>
              <mute>false</mute>
        </UserContext>
</contexts>
</events>
```

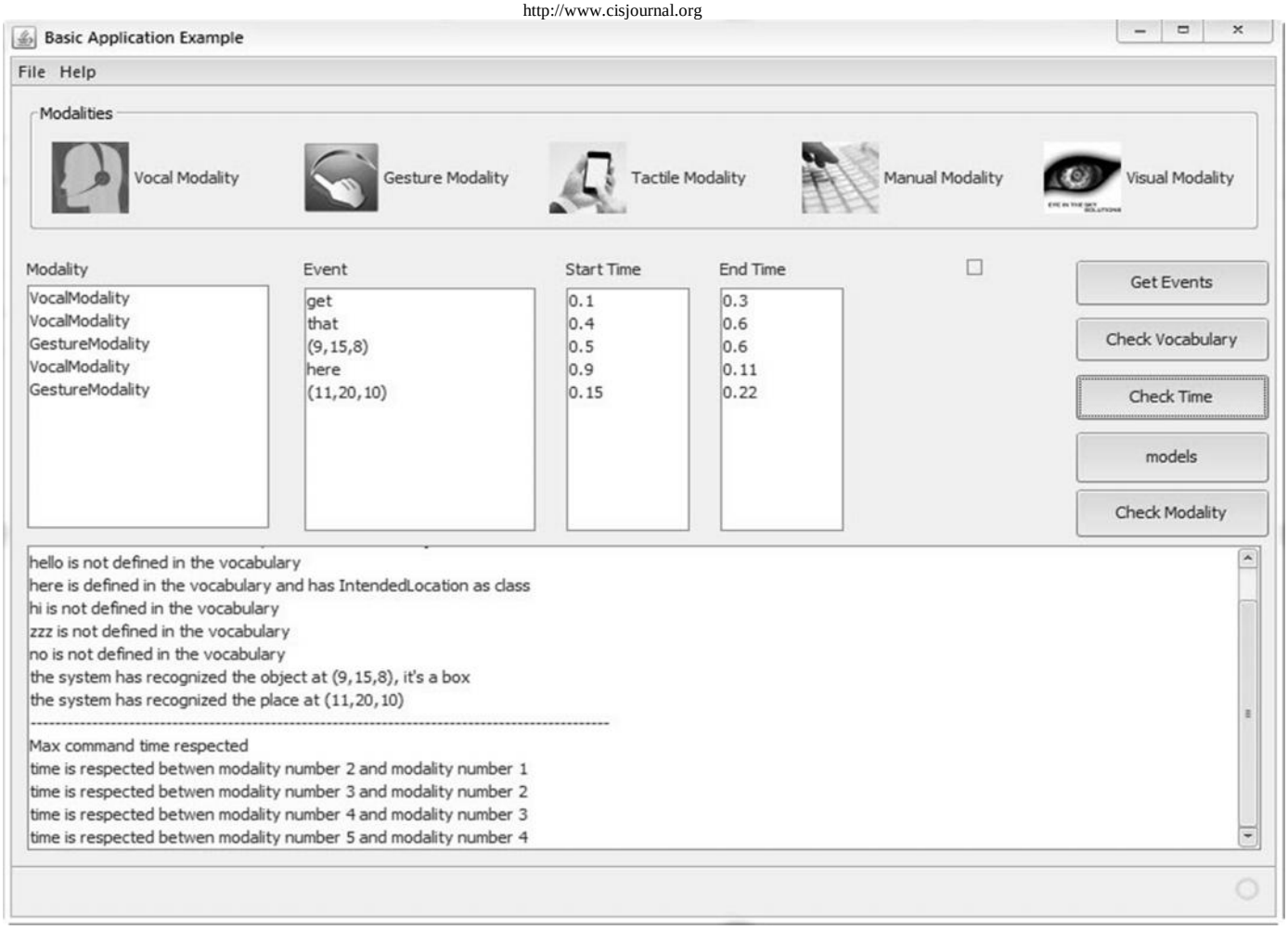

Figure 13: a screenshot of the tool

```
OWLOntology ont;
OWLOntologyManager man = OWLManager.createOWLOntologyManager();
String path = "file:/Users/Ahmad/Documents/NetBeansProjects/FusionSystem/EnvironmentOnto.rdf-xml.owl";
ont = man.loadOntologyFromOntologyDocument(IRI.create(path));
System.out.println("Loaded: " + ont.getOntologyID());

OWLModel owlModel = ProtegeOWL.createJenaOWLModelFromURI(path);
String rule1 ="Modality(?m) ^ Noise(?l) ^ hasNoise(?m, ?l) ^ hasNoiseLevel(?l, ?level) "
        + "^ swrlb:greaterThan(?level, 6) -> sqwrl:selectDistinct(?m)";
String rule2 ="Modality(?m) ^ Light(?l) ^ hasLight(?m, ?l) ^ hasLightnessLevel(?l, ?level) "
        + "^ swrlb:lessThan(?level, 4) -> sqwrl:selectDistinct(?m)";
String rule3 ="Modality(?m) ^ Handicap(?h) ^ hasUserContext(?m, ?h) ^ hasHandicap(?h, ?han) "
        + "^ swrlb:equal(?han, 1) -> sqwrl:selectDistinct(?m)";
String rule4 ="Modality(?m) ^ LocationContext(?loc) ^ hasLocationContext(?m, ?loc) "
        + "^ isAtRoad(?loc, ?loca) ^ swrlb:equal(?loca, 1) -> sqwrl:selectDistinct(?m)";
SQWRLQueryEngine queryEngine = SQWRLQueryEngineFactory.create(owlModel);

String qname1 = "Rule1";
queryEngine.createSQWRLQuery(qname1, rule1);
SQWRLResult result1 = queryEngine.runSQWRLQuery(qname1);

String qname2 = "Rule2";
queryEngine.createSQWRLQuery(qname2, rule2);
SQWRLResult result2 = queryEngine.runSQWRLQuery(qname2);

String qname3 = "Rule3";
queryEngine.createSQWRLQuery(qname3, rule3);
SQWRLResult result3 = queryEngine.runSQWRLQuery(qname3);

String qname4 = "Rule4";
queryEngine.createSQWRLQuery(qname4, rule4);
SQWRLResult result4 = queryEngine.runSQWRLQuery(qname4);
```

Fig 14: a screenshot of the code for modalities selection

Fig 15: a screenshot of the execution of code in **Figure 17**

Figure 13 represents the verification of different preconditions in the tool. This form was developed to see how the algorithm works. As shown; the recognized events are presented as lists. Each event is associated to a modality, start time, end time.

Fig 15 represents the execution result of the code in **Fig** 14 with the modalities that must be disabled depending on the context received from the environment. In reality, these modalities are instances in the ontology. Each instance belongs to a class of modalities. Knowing these instances, we can deduce what modality should be stopped or not.

Fig 16 is the real application that can be seen by a user, it contains different tactile buttons and different modalities. As shown, the system was able to recognize the model 08, so the command get that(pen) here (11, 20, 10) was identified as a user command.

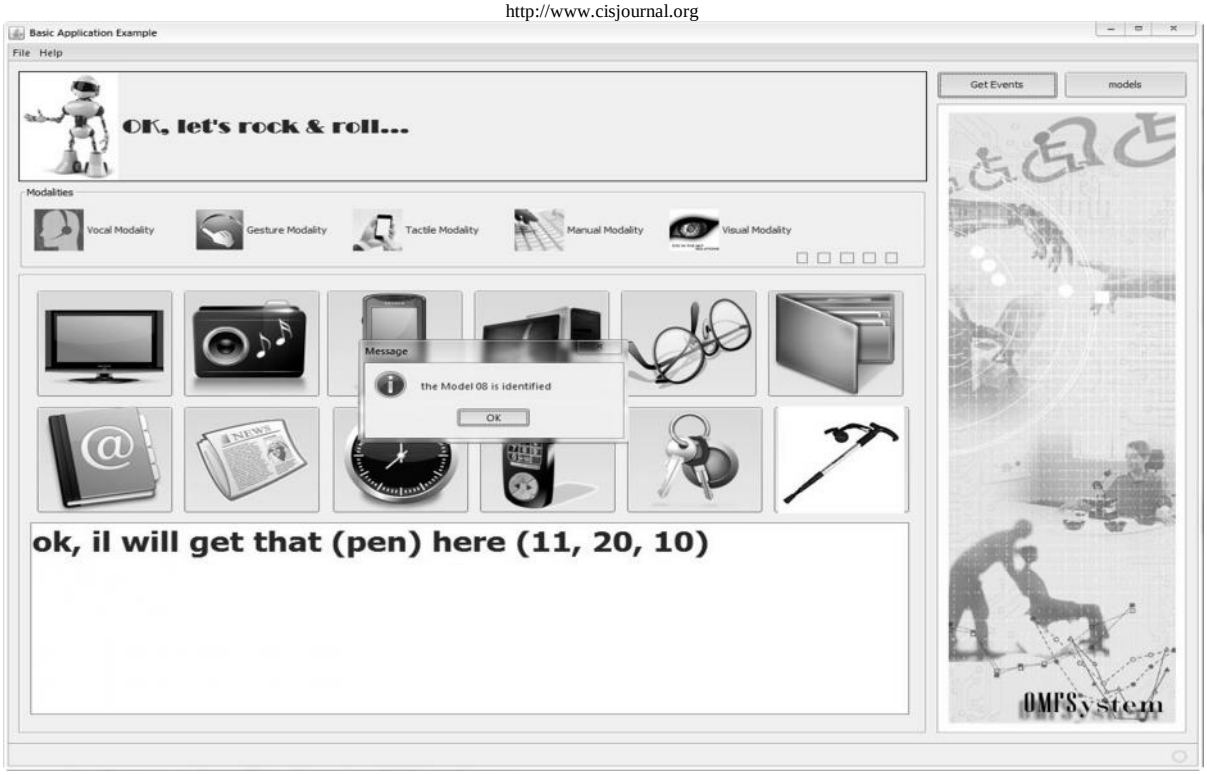

Fig 16: a screenshot of the real application intended to a user

6. CONCLUSION

In this paper, we presented an architecture that is very useful in a multimodal fusion system. We developed a system for multimodal fusion that allows multimodal interaction. In this interaction architecture several natural input modes (speech, pen, touch, hand gestures, eye movement, head and body movements) can be investigated. They are ultimately aiming at intelligent systems that are aware of the context and user needs. In this solution, the fusion engine is based on mechanisms of preconditions. The ontology is designed by defining a set of concepts, object properties, data properties and instances, taking into account special environmental and user contexts. We defined a fusion algorithm that take into account the vocabulary, the order of events and the temporal aspect for each of them. A prototype was made using OWL API and Java. This prototype is a functional validation of the approach. It shows that the proposed solution is applicable in a real environment. The events are verified according to contexts and the preconditions defined for fusion. The creation of the ontology made the description of the environment, the choice of modalities according to an environmental and user context easier by declaring all constraints and semantic relations between different concepts and by using SQWRL queries. To ensure the consistency of the ontology many SWRL rules were created. The ontology and its description language (OWL) are based on W3C standards which makes the proposed solution a formal solution.

This paper adopts an approach that take into account the description of environment and its constraints. The adoption of this system will facilitate the work of a multimodal interaction system by giving it the most meaningful combinations of events.

ACKNOWLEDGEMENT

This work has been made possible by the funding awarded by the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

[1] R. A. Bolt, ""Put-that-there": Voice and gesture at the graphics interface ACMSIGGRAPH Computer Graphics, v.14 n.3, p.262-270, July 1980

[2] H. J. Nock, Iyengar, G., and Neti, C., "Assessing Face and Speech Consistency for Monologue Detection in Video," Proceedings of ACM Multimedia, Juan-les-Pins, France, , 2002.

[3] U. Meier, Stiefelhagen, R., Yang, J., and Weibel, A., , "Towards Unrestricted Lip Reading,," International Journal of Pattern Recognition and Artificial Intelligence, vol. 14, no. 5, pp. , vol. 571-585, , 2000

[4] G. J. Wolff, Prasad, K.V., Stork D.G., and Hennecke, M., "Lipreading by neural networks: visual processing, learning and sensory integration," Proc. of Neural Information Proc. Sys. NIPS-6, Cowan, J., Tesauro, G., and Alspector, J., eds., pp. 1027-1034, 1994.

[5] J. Aleotti, Bottazzi, S., Caselli, S., and Reggiani, M., "A multimodal user interface for remote object exploration in teleoperation systems," IARP International Workshop on Human Robot Interfaces Technologies and Applications, Frascati, Italy, 2002

[6] D. Béroule, " Management of time distorsions through rough coincidence detection," Proceedings of EuroSpeech, pp. 454-457, 1989.

[7] B.-S. Shin, et al., "Wearable multimodal interface for helping visually handicapped persons," presented at the 16th international conference on artificial reality and telexistence, Hangzhou, China, 2006.

[8] R. Raisamo, et al., "Testing usability of multimodal applications with visually impaired children," IEE, Institute of Electrical and Electronics Engineers Computer Society, vol. 13, pp. 70-76, 2006.

[9] J. Lai, et al., "Examining modality usage in a conversational multimodal application for mobile e-mail access," International Journal of Speech Technology, vol. 10, pp. 17-30, 2007.

[10] M. Debevc, et al., "Accessible multimodal Web pages with sign language translations for deaf and hard of hearing users," presented at the DEXA 2009, 20th International Workshop on Database and Expert Systems Application, Linz, Austria, 2009.

[11] P. R. Cohen, Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. , "QuickSet: multimodal interaction for distributed applications. In Proceedings of the Fifth ACM international Conference on Multimedia. MULTIMEDIA '97. ACM, New York, NY, 31-40.," 1997.

[12] M. Johnston, Cohen, P. R., McGee, D., Oviatt, S. L., Pittman, J. A., and Smith, I., "Unification-based multimodal parsing," Proceedings of the 17th International Conference on Computational Linguistics, ACL Press, pp. 624-630, 1998.

[13] C. Elting, Strube, M., Moehler, G., Rapp, S., and Williams, J., "The Use of Multimodality within the EMBASSI System," M&C2002 - Usability Engineering Multimodaler Interaktionsformen Workshop, Hamburg, Germany, 2002.

[14] R. P. Engel, Norbert. , "Modality Fusion.SmartKom: Foundations of Multimodal Dialogue Systems. Springer, Berlin.," 2006.

[15] C. S. Barboni E., Navarre D., Palanque P. , "Model-Based Engineering of Widgets, User Applications and Servers Compliant with ARINC 661 Specification. Design Specification and Verification of Interactive Systems (DSV-IS 2006), LNCS, pp. 25-38. ," 2006.

[16] B. Dumas, et al., "HephaisTK: a toolkit for rapid prototyping of multimodal interfaces," presented at the Proceedings of the 2009 international conference on Multimodal interfaces, Cambridge, Massachusetts, USA, 2009.

[17] L. Wu, Oviatt, S., and Cohen, P. R., "Multimodal Integration - A Statistical View," In IEEE Transactions on Multimedia, vol. 1, no. 4, 1999. pp. 334-341., 1999.

[18] T. D. C. Little, Ghafoor, A., Chen, C.Y.R., Chang, C.S., Berra, P.B. , " Multimedia Synchronization," IEEE Data Engineering Bulletin, vol. 14, pp. 26-35, Sept. 1991 1991.

[19] T. R. Henry, Hudson, S.E., Newell, G.L.. , " Integrating Gesture and Snapping into a User Interface Toolkit, in Proc," presented at the Symposium on User Interface Software and Technology 1990.

[20] L. Nigay, Coutaz, J., Salber, D. MATIS, " A multimodal airline travel information system," Feb. 1993 1993.

[21] http://tt.stanford.edu/.

[22] http://www.w3.org/TR/owl-guide/.

[23] B. P. Evren Sirin, Bernardo Cuenca Grau , Aditya Kalyanpur , Yarden Katz "Pellet: A Practical OWL-DL Reasoner," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 5, pp. 51-53, 2007.

[24] http://www.jessrules.com/jess/charlemagne.shtml.