Specification and interpretation of multimodal dialogue models for human-robot interaction

Luis A. Pineda

Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS)
Universidad Nacional Autónoma de México, UNAM
Ciudad Universitaria, Coyoacán, México, D.F. CP 04510
Tel. +52 55 5622-3618, Fax. +52 55 5622-3620
e-mail: luis@leibniz.iimas.unam.mx

Abstract. In this paper a conceptual model for the design and construction of interactive multimodal systems is presented. This model is based on a representational language for the specification of dialogue models and its associated program interpreter. Dialogues models are domain and modality independent conversational schemes characterizing the dynamic of a multimodal interaction, and domain specific applications are represented as sets of dialogue models. Dialogue models are specified in terms of the intentions and actions, expressed and performed, in conversation situations, and represent the context for multimodal interpretation. In this paper, the language for representing dialogue models and the program interpreter are described in detail. The model is implemented as a multi-agent environment in which there is a main agent embedding the dialogue model's interpreter, the dialogue manager, and a specialized agent for each input and output modality. The theory has been tested with a simple application supporting spoken natural language input and output, graphics and video output, and also the motor action of a robot, which is also considered as an output modality. The present model has been implemented in the robot Golem, which has been widely demonstrated in Mexico, with a very positive response.

Key words: Dialogue managers, dialogue models, context representation, multimodal conversational systems, multimodal interpretation, multimodal presentations.

1 Introduction

Human-computer interaction follows protocols involving the expression and interpretation of intentions, and the execution of actions that satisfy such intentions.

There is a great range of flexibility in such protocols, from the very rigid and deterministic schemes involved in menu-based interaction to the rich and flexible patterns of some natural language conversation, like the so-called practical dialogues (Allen et al., 2000, 2001). Interactive protocols can also involve a range of modalities, from the textual to the full multimodal interaction involving spoken language, pointing actions and vision for the input, for instance, and spoken language, the display of images and videos, and also motor action of physical devices, like mobile robots, governed through the interface, for the output. The flexibility of multimodal interactive systems involves also the issues of coordination and reliability: the greater the flexibility in the interaction protocols and the range of modalities, the greater the need to make explicit provisions to coordinate information expressed in different modalities, and also to make the system robust. Another consideration is the need to establish a clear demarcation and modularity between interface issues and the application content proper. In this paper we address the problem of how to specify and construct flexible multimodal interactive systems, focusing in input and output intentions and actions, in a simple and reliable way.

The central tenet in the present approach is that the interpretation of expressions representing intentions in multimodal dialogues, and also the actions performed by agents as a consequence of understanding such intentions, are context dependent processes. Tasks oriented conversations, aimed to solve specific problems through a cooperative interaction, follow schematic protocols that can be construed as sequences of conversational situations; within the context of such protocols only a very limited number of intentions are meaningful in relation to the context in a given situation, and only a small set of actions are relevant to achieve the goals of the task in that particular situation. Consequently, it is possible to think of conversational situations in terms of the set of expected intentions that can be expressed by other agents in the situation, the set of possible actions that ought to be performed by a conversational agent as a response to the recognition of one such expected intention, and on the conversational situation that is reached as a consequence of performing the corresponding an action. We pose that the set of conversational situations, with the actual expressed intentions and performed actions, visited during a specific conversation, constitutes the conversational context, and the instantiation of the corresponding dialogue model is a representation of such conversational context.

In the present framework we distinguish between propositional and predicative dialogue models; in the former kind each expected intention is a concrete proposition, and is represented by a propositional constant or a grounded predicate. Manu-based applications, for instance, in which every time the human-user makes a choice in a context in which all the parameters of the subsequent action are already determined, is a propositional dialogue model. In predicative dialogue models, on the other hand, an intention type with its parameters needs to be recognized from the external input, and the result of the interpretation process is the actual concrete intention that ought to be satisfied by the agent; accordingly, predicative intentions are represented through expressions including variables and the functional abstractor. This is, intentions are represented by functions, which are bounded and reduced by functional application in the interpretation process. In a natural language conversation situation in which the computational agent expects an information request or an action directive expressed by the human-user, with the

corresponding propositional content, for instance, the information request and the action directive are the intention types, while the arguments correspond to the propositional content stating what information should be provided or what action should be performed. In the present paper we pose the hypothesis that task-oriented conversations or practical dialogues can be modeled through predicative dialogue models.

Propositional dialogue models can be represented explicitly through networks defined in advanced through analysis; however, although this strategy has important applications, it renders very rigid interaction protocols; on the other extreme, an unrestricted predicative dialogue model can be represented implicitly through models that retain an "information state" and predict the potential "moves" that can be performed in such a state, as in the TRINDI project (Larsson and Traum, 2000) and the DIPPER architecture (Bos *et* al, 2003), with the subsequent complexity and computational cost. The model presented here is a compromise between these two extremes; although a specific graph structure is preserved explicitly for the representation of propositional dialogue models, part of the graph can be represented implicitly through the specification of functions that map local and global information into concrete expressions; these functions are evaluated on the fly and the actual concrete intentions and actions that result from functional application, and also the situations that are reached as a consequence of such actions, can be determined dynamically.

Modeling the conversational context through dialogue models suggest a heterarquic processing architecture in a natural way: modality specific input information is processed in a bottom-up fashion, but the representation of the target expected intentions that need to be recognized are provided top-down from the dialogue model. This is relevant for reducing significantly the ambiguity that needs to be resolved in the interpretation process, like the ambiguity involved in natural language interpretation. We pose that this is also relevant for reference resolution and the interpretation of vague expressions. Unlike approaches that aim to map the external information provided by modality and domain specific perceptual process into a context independent semantic representations, in the present approach every interpretation state has a finite and normally small number of target pragmatic representations representing the set of expected intentions in the conversational situation, and the goal of perceptual interpretation is to map the input information into the most likely expected intention in the interpretation situation.

This feature is also essential to make the system robust; the conversation can proceed only in case one expected intention is recognized in the input; in case the message cannot be mapped into one of these with a reasonable level of certainty, the agent needs to express that the input was not understood in relation to the conversational context, despite that the result of the perceptual process may be a meaningful representations, and the intention interpretation process needs to continue until a relevant intention is hypothesized by the agent. More generally, the dialogue manager needs to address explicitly the grounding problem: conversational obligations, like information requests or action directives, must be accepted or rejected, and communication failures must be fixed (Pineda *et al.*, 2007). In the present approach we explore the view that grounding behavior can also be modeled through dialogue models that are interpreted when such failures occur, and when

these are fixed, the context is restored and the conversation is resumed at the situation where the grounding failure first occurred.

Accordingly to this discussion, in the present framework interactive multimodal interaction is specified at the intentional level. This specification involves the definition of schematic protocols stated in terms of the intentions and actions expressed, interpreted and performed by both the human and the computer agent in the course of the interaction; listening, speaking, seeing and moving, are all considered context dependent intentional actions, which are performed as a consequence of interpreting the expressions representing the corresponding intentions. Also, multimodal content is referred to within the structures representing the dialogue models, but the content proper is stored independently of these structures, and it is retrieved and combined dynamically in the construction of multimodal presentations. The meet these desiderata, intentions are expressed in a common domain and modality independent representational language; also, the interpretation of such expressions is performed by an interpreter program, also domain and modality independent. Information concerning specific modalities and application content, on the other hand, is left to the semantics of the language, and basic expressions can be interpreted by domain and modality specific process.

In this paper we present this conceptual framework for the specification and implementation of multimodal conversational systems in detail. In Section 2 a language for the representation of conversational protocols at the level of intentions and actions is presented. This language is defined in the terms of a formalism that we called functional recursive transition networks, which augments recursive transition networks with functions labeling arcs and states. This formalism has a graphical representation which facilities greatly the specification and interpretation task. In these graphs, nodes represent conversational or interactive situations, and arcs are labeled by the intentions that need to be expressed to reach the corresponding situation, and by the actions that are performed when such situations are reached. The language supports the definitions of propositional dialogue models in which intentions and actions can be stated concretely through basic constants and grounded predicated but, in addition, a functional mechanism to specify intentions and actions in terms of the current situation, the current dialogue model, and the history of the interaction, is also defined. This is, the system supports a limited form of predicative dialogue models, increasing significantly the expressive power of the formalism with a limited additional computational cost. The language is illustrated with the graphical and formal expressions representing a dialogue model in a simple application domain, an example of the full specification of a dialogue model is avaliable in the project's web-page[1].

Next, the program interpreter is presented in detail in Section 3. This program interprets the dialogue models and the interpretation process runs hand in hand with the interaction of the computational agent with the world, and this process proceeds until the main dialogue model reaches its final situation. The multimodal platform has an agent for each modality supported by the system, and these agents are called upon when the semantics of an intention needs to be computed by the dialogue manager. For instance, in the case of spoken natural language input, the

[1]http://leibniz.iimas.unam.mx/~luis/golem/administrador.html

representation of an intention, or a set of possible intentions, that is expected in a particular situation is represented and interpreted by the dialogue manager, but speech recognition, and also lexical and syntactic processing, is encapsulated in the agent supporting the spoken language modality. Similarly for the output modalities. The full commented Prolog's code of a test version the interpreter is presented in the projet's web-page (see note one above).

In Section 4, a simple application is described. In this application a robot gives a tour to the visitors of our department and explains the posters of the projects that are currently being developed. The conversation is carried on in spoken Spanish, and in the course of the explanation, also in spoken Spanish, images and videos are displayed; in addition, the robot moves to the physical places where the corresponding posters are located. The spoken output, the display of images and videos, and the robot movement are all considered output modalities, and are treated by the same modality independent mechanisms.

The paper is concluded in Section 5 with a reflection of the potential and limitations of the present approach, and a discussion of some further lines of research.

2 Representation of dialogue models

In this section the specification for the functional recursive transition networks formalism for representing dialogue models is presented. In these networks nodes represent conversation situations and arcs the input intentions expressed by other agents and also the (output) actions, either linguistic, motor or multimodal, performed by the agent in the situation; we refer to the former as (input) speech acts (*SA*) and to the latter as (output) rhetorical acts (*RA*). Rhetorical acts are complex actions that are composed –and represented—in terms of a number of modality specific basic acts, and we refer to the representation of this complex object as multimodal rhetorical structure (*MRS*).

Situations are related to behaviors and modalities, and the language supports *listening*, *telling*, *error*, *final* and *recursive* situations. Each type of situation has a particular interpretation strategy; listening situations interpret input speech acts and perform an output rhetorical act; telling situations have an empty input, and their purpose is to perform an output rhetorical act only; error situations are reached every time the input cannot be mapped to an expected intention of the current situations, and its role is to execute a conversational protocol that has the goal of fixing the communication failure, restoring the context, and resuming interpretation of the situation where such a failure was originally found. Final situations, in turn, signal the end of the protocol. Recursive situations stand for full dialogue models, which are interpreted whenever a situation of this type is reached; there are no restrictions for the level of embedding of these situations, allowing great modeling flexibility; whenever the final situation of an embedded model is reached, the interpretation of current model is concluded and the model containing the corresponding recursive situation resumes execution. Reaching the final situation of the main dialogue model ends the conversation.

Every listening situation has a list of expected intentions that are meaningful in relation to the conversational context, and speech acts expressed by other agents are interpreted in relation to such a list in the current interpretation situation. So, the problem of speech act interpretation is posed generically as what is the most likely expected intention that is intended by the conversational partner given the properties of the actual message in relation to the current conversational context. In this approach, overt messages are taken as providing evidence for selecting one expected intention, and the intention selection process relies also on low level processes that take into account both the form and content of the available information. This specification can be a simple matching process, based on regular expressions, for the identification of propositional intentions, or a complex stochastic process involving intonation, and also lexical and syntactic form, to determine the type and content of a predicative intention[2].

The description of a situation involves also the specification of the rhetorical act that needs to be performed when one expected intention is recognized and the situation that will be reached when such an act is performed. Rhetorical acts can be stated as grounded propositions or as predicates including variables that need to be bound in the interpretation process. Rhetorical acts are thought of as a sequence of basic actions, perhaps performed in different modalities, so when the structure representing a rhetorical act is interpreted, all actions in the corresponding modalities are performed as a holistic "encapsulated" unit. For the specification of these complex units the formalism supports the definitions of rhetorical acts types. The definition of rhetorical acts follows loosely Rhetorical Structure Theory (*RST*), originally developed by Mann and Thompson (1988), and an instance of a rhetorical act type can be thought of as a multimodal "paragraph". This approach has also antecedents in the work of Feiner and McKeown (1993), Wahlster *et al.* (1993) and also Moore (1995), although the emphasis in those approaches was to plan the multimodal rhetorical structure for intelligent multimodal presentations. In the present approach, on the other hand, we are focused on the specification of multimodal rhetorical structures for practical applications in the context of multimodal conversations.

In the graphical representation nodes represent situations and arcs are labeled by pairs containing an input speech act and the output rhetorical act (i.e. *i-sa*:*o-rha*), where *i-sa* stands for the expected intention that needs to be recognized from the input in order to travel through the arc, and *o-rha* stands the output rhetorical that is

---

[2] The schema can also be thought of in term of the noise channel model and the Bayes theorem (e.g. Jurafsky, 2000). The set of expected intentions has an *a priori* probability to be selected in a conversational situation, the recognition process provides the probability that the messages expresses an expected intention given the form of the message proper, and the selected intention is the one that maximizes the product of these two probabilities; so, highly expected intentions can be selected even with limited and noisy information from the input, and also a highly informative message can select the right intention, even if it's *a priori* probability is quite low. Although in the current implementation we have not assigned probabilities to expected intentions or messages, the approach presented below can be thought of in terms of the Bayes model although in the limiting discrete case.

performed in the transition. More generally, a situation *s* is an abstract object with an input and an output parameter pair: $s(i_i{:}o_i, i_o{:}o_o)$. We call $i_i{:}o_i$ the *in* pair, and $i_o{:}o_o$ the *out* pair.

Fig. 1 is an instance of a dialogue model. This is the main dialogue model for an application in which a robot gives a guided tour, explaining the research areas and projects developed at our research department.

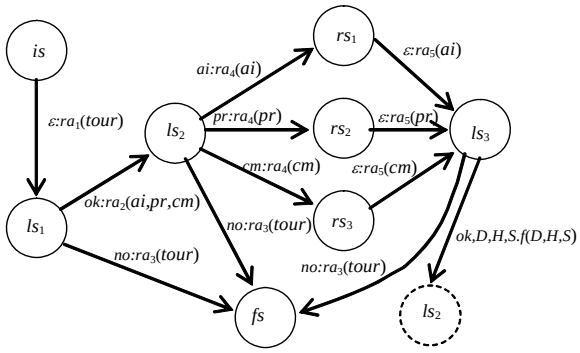

Fig. 1. The main dialogue model

This model has one initial telling situation *is*, three listening situations ls_1, ls_2 and ls_3, three recursive situations rs_1, rs_2 and rs_3 and the final situation *fs*. There is also an error situation (not shown in the diagram) that is reached whenever the input cannot be mapped to one expected intention. This particular dialogue model is defined in terms of five propositional expected intentions that have the following interpretations.

- *ai* = user wants to visit the artificial intelligence section of the department
- *pr* = user wants to visit the pattern recognition section of the department
- *ca* = user wants to visit the combinatorial analysis section of the department
- *ok* = user accepts current offer
- *no* = user declines current offer

In the interpretation process the spoken input is mapped into one of these intentions in the corresponding situations. In ls_1, for instance, the system needs to distinguish between whether the user is accepting or rejecting an offer (expressed by an instance of ra_1, as will be explained below) regardless the actual accepting expression (e.g. “yes”, “please”, “aja”, “yes, I do”, “yes, I would like to”, etc.) or the rejecting expression (e.g. “no”, “no, thanks”, “not now”, etc.). Similarly for situation ls_2 where the user needs to express which area he or she wishes to visit or whether he or she wishes to end the tour, and there are also an arbitrary number of ways to express these choices.

The dialogue model also specifies five propositional rhetorical act types ra_1, ra_2, ra_3, ra_4 and ra_5 in addition to a predicative rhetorical act ra_{20} (not shown in the diagram), which is specified through the function *f*, as will be explained below. Propositional acts may be named by propositional constants, or may be stated as

grounded predicated, as it is the case in the five acts in Fig. 1. In the example, ra_1 is an invitation, which has the word "tour" as a parameter; this act is generic and the parameter states the content of what is being offered. In this particular example, the system invites the user to do a tour through performing this act. User defined rhetorical acts are performed at the time the corresponding expression is interpreted by the dialogue manager. For instance, ra_1 is defined as a greeting expressed linguistically, the display of a welcoming picture, a linguistic introduction to what has been offer and, finally, a yes/no question asking whether the offer is accepted or not. As can be seen in Fig. 1, ra_1 is the rhetorical act of the out-pair of the initial telling situation, and also the rhetorical act of the in-pair of ls_1, and $ra_1(tour)$ is interpreted and performed when the dialogue model is started.

The act ra_2 is an offer to select a course of action among a predefined number of options, which are listed as the parameters, so $ra_2(ai, pr, ca)$ renders an offer to visit the corresponding areas; this is a multimodal act including a picture and its caption presenting the area, followed by the presentation of a textual menu with the aspects that can be visited (e.g. academic staff, research areas or research projects), followed by a verbal description of these options, and a question asking what aspect should be explained. The specification of the listing situation ls_2 includes the expected intentions, and the verbal answer input by the user needs to be mapped into one of these, regardless of the actual form of the expression.

In the graph there is also a rhetorical act specified as the function *f*. This function maps the current dialogue model, the history of the interaction and the current situation (the variables *D*, *H* and *S* respectively) into a concrete rhetorical act; in the interpretation of this arc the function is first reduced, and the resulting rhetorical act is performed. This facility is used to define rhetorical acts on the fly; for instance, if the user already visited the artificial intelligence section, the evaluation of *f* would result in the expression $ra_{20}(pr, ac)$; this rhetorical act is like ra_2 but without the input picture and verbal caption that should displayed and performed only the first time the situation ls_2 is reached, and its interpretation would only render an offer of the two remaining options.

Recursive situations are interpreted as full dialogue models, which are called when the corresponding situations are reached. The dialogue model for the situation rs_1 of the main dialogue model in Fig. 1 is illustrated in Fig. 2.

This model exemplifies, in addition to the previous one, a function *g* which depends on the current dialogue model *D* and the conversations history *H*; this function is evaluated when this dialogue model is started and its value is the rhetorical act ma_1; this act renders a multimodal paragraph including the display of a picture but, in addition, it contains a basic motor rhetorical act *ma*(*from*, *to*); performing this latter act produces that the robot moves from its previous location (i.e. the situation that the robot was placed when the calling dialogue model was executed) to the location it should move to explain the current model. This facility shows that any intentional action performed by the agent can be model as rhetorical act, independently of the modality in which actual message is presented, or the action is performed. In this model, the act ra_3 is also multimodal, and its interpretation renders a paragraph involving spoken language and the display of a picture, as specified by the corresponding parameters.

A Dialogue model is specified through a three place predicates of the form *diag_mod*(*ID*, *S*, *R*) where *ID* is a unique identifier for each dialogue model, *S* is the list of situations of the model, and *R* is the list of rhetorical acts in the model; situations and rhetorical acts are specified, in turn, in terms of attribute-value pairs.

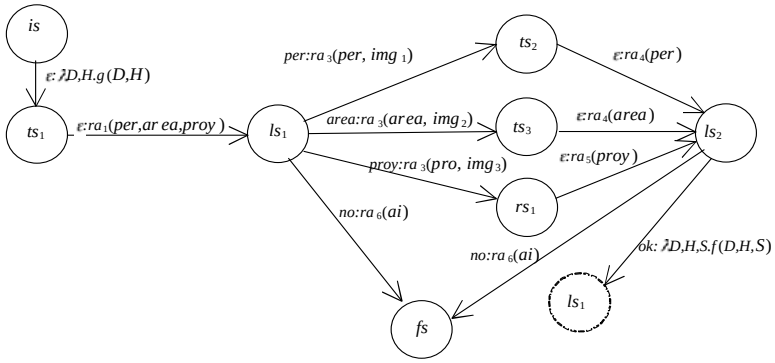

Fig. 2. An embedded dialogue model

Each situation has the following main attributes: *id*, *type*, *in-pairs* and *out-pairs*. The values for these attributes are the situation's unique identifier, its type, the list of in-pairs from which the situation can be reached, and the out-pairs of the situation. There is no limit to the situations that can be included in a dialogue model, neither to the number of in and out pairs that a situation can have. Also, identifiers are local to the dialogue model, and have no scope in the interpretation of other models.

The specification of situation ls_3 of the main dialogue model, for instance, is shown in Fig. 3.

```
[   id ==> ls_3,
        type ==> listening,
        in_pairs ==> [   rs_1 => empty:ra_5(ai),
                         rs_2 => empty:ra_5(pr),
                         rs_3 => empty:ra_5(ca)],
        out_pairs==>[ok:(lambda([D,H,S],
                           f(D,H,S)),[[ai,pr,ca], ra_2_0]) => ls_2,
                         no:ra_3(tour) => fs],
        expected_intentions ==> [ok, no]
        ]
```

Fig. 3. Specification of a situation

In Fig. 3 attributes and values are related through the operator "==>" and input and out-pairs have the form s_p => *i*:*o* and *i*:*o* => s_n respectively, where s_p and s_n

stand for previous and next situation. The specification is stated in the actual Prolog notation, and capitals stand for variables and lower-case letters for predicates and constant symbols. The output rhetorical act associated to the *ok* speech act is specified through the function *f*; this function has as the dialogue model identifier *D* and the interaction history *H*, in addition to a situation argument *S*, as its arguments. The first two parameters are accessible directly to the dialogue manager, as will be explained below, and only the local parameter needs to be specified for functional application; this is, in turn, a list of arguments, which in the present example is the two elements list *[[ai, pr, ca], ra_2_0]*, which in this case are a list of arguments and a predicate. The function *f* is defined explicitly by the user, as will be explained below, and it computes the difference *L* between a given options list (i.e. [*ai*, *pr*, *ca*]) and the options already visited (available in the interaction's history *H*), and returns as its value the predicate *ra_2_0*(*L*). This rhetorical act is then executed rendering the corresponding offer and the situation ls_2 is reached again.

In the specification, the *in–pairs* attribute is optional, and can be used for checking coherence between the situations, except for recursive and error situations for which this argument is obligatory; in former case the input speech act of the previous situation determines the dialogue model to be loaded and interpreted in the recursive situation. In the case of error situations, which are reached whenever none of the expected intentions can be chosen from the external linguistic input, the communication failure should be fixed without altering the conversational context; for this, the in-pair of the error situation is also its out-pair, as specified in Fig. 4. As can be seen, the listening situation that is reached from the error situation is also its previous one; in addition, the interpreter passes the in-pair of the situation where the failure occurred as the in-pair of the error situation, and the net effect of the mechanism is that the in pair of the listening situation where a failure occurred is preserved after the error situations has been visited. This is an instance of how in and out pairs can be used to keep the conversation coherent.

```
[   id ==> unexpected_speech_act,
    type ==> error,
    in_pairs ==> [In_Pair],
    out_pairs==>[In_Pair=>Previous_Situation]
]
```

Fig. 4. Specification of an error situation

In a more elaborate setting, a speech act has, in addition to its future effect on the dialogue (i.e. its forward functions), some backward functions that depend on previous discourse; so, information for anaphoric resolution, for instance, could be available from the *in-pair*. However, we left the study of an explicit handling of backwards functions of dialogue acts and dialogue coherence for further research.

The rhetorical structure *R*, specifying the output act is similarly stated. For instance, the specification of the act ra_2 of the main dialogue model is shown in Fig. 5 as follows:

```
[   id ==> ra_2,
        type ==> ofert-introduction,
        pars ==> [Topic],
        rht_acts ==> [display('foto_depto\.jpg'),
    caption(personal),
        display('dcc_areas\.jpg'),
        introduction(Topic,'el departamento','las areas de '),
        open-option-what('visitar','area')]
```

]

Fig. 5. Specification of a rhetorical structure

As before, the operator "==>" relates attributes with their corresponding values; the value of the *id* attribute is a unique identifier for the rhetorical act within the dialogue model; *type* identifies user's defined multimodal paragraph, and the value of *pars* is the arguments list of the paragraph; finally, the value of the *rht_acts* is a list of modality specific basic rhetorical acts composing the present *MRS*. These basic acts are defined in advanced for the whole set of dialogue models of a conversational domain. In the example, the arguments of the basic acts can be defined in the argument list of the whole rhetorical structure (the value of the attribute *pars*), although these can also be constants that have a direct interpretation by the modality specific rendering process. For instance, the basic act *display* renders its image parameter on the screen, *caption* renders a text associated to the image's caption identifier as spoken output; *introduction* is a textual template, also rendered as spoken output, that produces a sentence in which the subject is its second argument, and the predicate is headed by the third argument and modified by the conjoined terms in the topic's argument list. So, if *Topic* is the list [*ai, pr, ca*] the interpretation of this basic rhetorical act renders "el departamento tiene las áreas de inteligencia artificial, reconocimiento de patrones y análisis combinatorio" (*the department has the areas of artificial intelligence, pattern recognition and combinatorial analysis*). Finally, the last basic act is also a textual template that produces the question "qué area quieres visitar" (*what area would you like to visit*), which is also rendered as spoken output. Finally, every dialogue model needs the specification of an *error* rhetorical act, which is performed as the output rhetorical act of the error situation, as shown on the project's web-page (see note one above).

The application content, such as texts, images, videos, etc., is referred to through the parameters of the basic acts, but content proper is stored in an external memory structure, and the code to retrieve such information is stated in the modality specific agent that performs such basic acts. However, the rhetorical structure as whole organizes the multimodal information a "multimodal paragraph", and when the rhetorical structure is interpreted, the multimodal information is rendered as an indivisible act. In this way, issues about content are completely detached from the multimodal interpretation process, which is focused on the interpretation and performance of speech and rhetorical acts at the intentional level.

Finally, a conversational domain is specified as a list of dialogue models, each defined through a *dig_mod* predicate, as well as the set of basic rhetorical acts that used in the application, as will be explained below. The full specification of the main dialogue model in Fig. s 1 is shown in the project's web-page (see note one above). The dialogue model in Fig. 2 is similarly defined.

3 The Dialogue Manager

Dialogue models are interpreted by an interpreter program, which is called "the dialogue manager" as illustrated in Fig. 6. The objects of interpretation are the dialogue models directly, and a situation is interpreted in each interpretation cycle. Each situation type has its own interpretation strategy, in which the *in* and *out* pairs are evaluated, and the next situation is selected for interpretation.

The interpreter first loads the *diag_mod* list, which can be considered as the evaluation environment for the dialogue manager. The interpreter has three main utilities: *get_dialogue*, *get_structure* and *get_feature_value*.

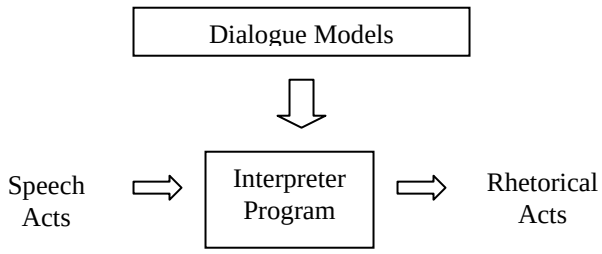

Fig. 6. Dialogue Manager

The first retrieves the list of situations and the list of rhetorical acts of a dialogue model given its unique identifier; the second retrieves a situation from a dialogue model given the situation's id, and the third retrieves the value of a feature from a list of feature-value pairs. In particular, whenever a recursive situation is called, the embedded model is retrieved from the models' list, and the control is past to its initial situation of such model, and only when the final situation of this latter model is reached, the interpretation of the recursive situation of the embedding model is resumed. In this way, the interpreter implements a stack strategy for the interpretation of dialogue models, and its computational power is equivalent to that of context free grammars.

Dialogue models are schematic representations of interaction protocols, and several instance of the same dialogue can arise in the course of a conversation; for this, every time a dialogue model is called upon an instance of the scheme is created, and the actual specification of the dialogue model is never altered during the interpretation process. For this, whenever a situation is interpreted, a new instance of the situation (i.e. with new variables) is created; similarly for instances of the corresponding rhetorical acts.

Dialogue models represent generic *a priori* interpretation context for speech acts, but in addition, a specific conversational context is created incrementally during each conversation. This context is recalled in the conversational history, which is defined as the list of grounded situation instances of the form $model\_id{:}s(i_i{:}o_i,\ i_o{:}o_o)$. This list is available in the interpretation of every situation along the interaction, and its content is accessible through the definition of functions, as was exemplified above. More generally, this list codifies all references made during the conversation up to the current situation, and the dialogue model within which each situation was grounded.

The interpreter itself has two main parts: the logic to explore the dialogue models with a stack based discipline, and a specialized procedure to interpret instances of each kind of situation. Each one of these latter procedures has a particular interpretation discipline according to the situation's type, and concludes with the execution of the procedure *perform_rht_act*; this procedure first recovers the rhetorical structure of the rhetorical act to be performed from the current dialogue model (e.g. the structure associated to the speech act recognized from the input in listening situations); then creates an instance of each basic act in the list, binds its arguments, and performs each act by sending a message to the corresponding modality specific rendering agent. The full Prolog´s code of a test

version of the interpreter program (in which only rhetorical acts can be specified through functions) is presented in the project's wab-page (see note one above).

4 System architecture and implementation

This theory has been tested with the implementation of a conversational robot named *Golem*, a *RWI Magellan Pro*, which is able to sustain a simple conversation is spoken Spanish; in addition, the answers and explanations provided by the robot are supported with the displays of menus, texts, pictures and videos on a screen, and the robot's movement is also modeled as a basic rhetorical act defined within the context of a rhetorical structure in the dialogue model. The system has a programming interface through which it is possible to read the state of every sensor and to command the robot's motor behavior.

The robot's multimodal behavior is modeled through a set dialogue models defined for the application domain. The central component for the multimodal interaction is the dialogue model's interpreterwhich directs the robot's behavior in terms of the intentions and actions associated to the conversational situations. As was mentioned, in the current application the robot guides a visit to a poster's session about the research projects developed at the Department of Computer Science at IIMAS, UNAM. The dialogue models represent both the conversational protocols to carry on with the visit, and index the conceptual content that can be referred to through the conversation. As a part of the exercise a number of posters about the projects were built; this task was developed by academic assistants and students familiar with the structure of dialogue models; the task focused on a number of interviews to the department's research staff, who provided the content to be explained, including texts, pictures and videos to be displayed within the multimodal explanations.

For the computation implementation the Open Agent Architecture (OAA) was adopted[3]. In this architecture it is possible to associate diverse computational processes to "agents" that form a part of the computational agent as a whole. These processes can be defined in different programming languages, like Prologo, C, C++, Perl, Java, etc., and in diverse operating systems, like linux and windows, as it is the case in the present implementation. The main agent runs the dialogue manager and subordinates all other agents; there is a modality specific agent for every input and output modality: the speech recognition system, an in house system built with Sphinx[4] and the Corpus DIMEx100[5], an agent that interprets the recovered text in terms of the intentional expectations associated to the current interpretation situation, the speech synthesizer, the agents for the display of images and videos, and the agent that controls the robots movement directly. The system's architecture is illustrated in Fig. 7.

[3] http://www.ai.sri.com/ ~oaa
[4] http://cmusphinx.sourceforge.net/html/cmusphinx.php
[5] http://leibniz.iimas.unam.mx/~luis/DIME/CORPUS-DIMEx100.html

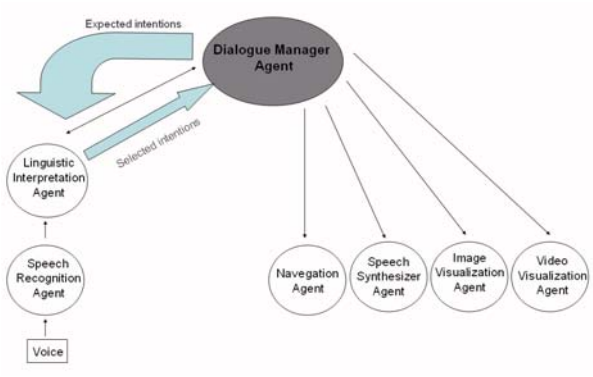

Fig. 7. Agent's architecture of the robot Golem

A video of the project (in Spanish) can be seen in the project's home page[6].

5 Discussion and further work

In this paper we have introduced the notion of dialogue models for the representation of multimodal interpretation contexts and its formalization through their associated program interpreter, which we have also equated with a generic multimodal dialogue manager. Dialogue models are also representations of application domains at the intentional level; this is, in terms of conversational situations with their associated expected intentions and actions, which are represented in a modality independent fashion. In addition, the model supports a strict separation of the interpretation context and the content expressed through a multimodal interaction: while the interpretation context is a modality and content independent pragmatic representation, the content evoked through the parameters of modality specific rhetorical acts is retrieved from memory specialized modality specific processes.

The model permits the specification and implementation of applications in a simple declarative way, and it has been tested with an application involving the interaction with a mobile robot through spoken Spanish, where the answers and explanations given by the robot are supported by texts, images and videos, and the robot movement is also handled as a standard
output modality.

The present approach is aimed for modeling collaborative task oriented multimodal conversations in specific domains, and we pose that a large class of interesting multimodal applications belong to this class. In potential applications the interaction with multimodal devices can be thought of in terms of intentions and actions types embedded in conversational protocols. These protocols can then be represented through dialogue models in a declarative way, and the dialogue is handle by a generic domain independent interpreter program.

The current model handles propositional dialogue models and also a limited class on predicative models, in which rhetorical acts can be stated through functions. The model is also restricted in that only the speech modality is defined for the input; for this, we are developing the interpreter to support pointing actions and some vision facilities, although within the same general architecture.

[6]http://leibniz.iimas.unam.mx/~luis/golem/golemenlosmedios.html

However, the concept of dialogue model and its associated dialogue manager interpreter are modality independent and, in principle, any input and output modality can be incorporated in practical applications.

The current implementation is also limited in that expected intentions are represented though constants, and the linguistic input is only used to select one expected intention among the set of expected intentions in the situation; however, we are also extending the model to permit the expression of intention types whose arguments need to be retrieved from the input message; this facility will permit to extend considerable the kind of dialogues that can be modeled, and we expect this extended set to be equivalent to the set of practical dialogues. Also, although the current heterarquic model can be thought in terms of a Bayesian model in the limiting discrete case, we plan to incorporate *a priori* probabilities to the expected intentions or intention types. In order to define these probabilities a corpus in the application domain will be collected and tagged. We will also plan to incorporate a probabilistic parser to the system, to get a number of possible interpretations of the linguistic input with their corresponding arguments, in order to rank the potential interpretations according to the product of the a priori probability of each intention and the probability of the interpretation provided by the probabilistic parser, extending in this way the current interpretation model based on regular expressions. This strategy will allow us to implement a system of intention recognition with a symbolic component combined with a stochastic process. We hope that the combination of the structure of dialogue models with the probabilities associated to intentions and syntactic interpretations will provide a general robust methodology for the specification and implementation of very flexible multimodal interactive systems.

Acknowledgements

The authors gratefully acknowledge the participation of the members of the DIME group at IIMAS, UNAM, specially to Ivan Meza, Paty Pérez Pavón and Wendy Aguilar, for their support in the implementation work; also to James Allen for useful comments. The author also acknowledges the support of NSF/CONACyT grants C092 and 39380, and PAPIIT-UNAM grants IN-111700 and IN-121206.

References

Allen, J.F., & Core, M. (1997). Draft of DAMSL: Dialogue Act Markup in Several Layers, Technical Report, The Multiparty Discourse Group. University of Rochester, Rochester, USA.

Allen, J.F.; Byron, D. K.; Dzikovska, M.; Ferguson, G.; Galescu, L. & Stent, A. (2000). An Architecture for a generic dialogue shell. Natural Language Engineering, 6(34):213–228.

Bos, J., Klein, E., Lemon, O., Oka, T. (2003). DIPPER: Description and formalization of an Information-State Update dialogue system architecture, Proc. 4th SIGdial Workshop on Discourse and Dialogue, pp. 115—124, 2003.

Feiner, S. K. and McKeown, K. R. (1993). Automating the Generation of Coordinated Multimedia Explanations. In Intelligent Multimedia Interfaces, edited by Mark T. Maybury, pp. 117--138. AAAI Press / The MIT Press.

Jurafsky, D., Martin, J. H. (2000). Speech and Language Processing, Prentice Hall, New Jersey.

Larsson, S. and Traum, D. (2000), Information state and dialogue management in the TRINDI Dialogue Move Engine Toolkit, *Natural Language Engineering* 6 (3-4): 323—340.

Mann, W. C. and Thompson. (1988). S. Rhetorical Structure Theory: Towards a functional theory of text organization, Text 8(3), pp. 243—281.

Moore, J. D. (1995). Participating in Explanatory Dialogues, A Bradford Book, The MIT Press.

L. Pineda, V. Estrada, S. Coria y J. Allen. (2007). The obligations and common ground structure of practical dialogues, *Revista Iberoamericana de Inteligencia Artificial*, Vol. 11 (36), pp. 9-17.

Wahlster, W., André, E., Finkler, W. and Rist, T. (1993). Plan-based integration of natural language and graphics generation, Artificial Intelligence 63, pp. 387—427.