A two-headed architecture for intelligent multimedia man-machine interaction[1]

Paulo Quaresma[2] and José Gabriel Lopes

Centro de Inteligência Artificial, UNINOVA
Quinta da Torre, 2825 Monte da Caparica Portugal
email: {pq|gpl}@fct.unl.pt

Abstract

In this paper a two-headed architecture designed for handling intelligent multimedia man-machine interaction is described. In this work we are mainly interested in interactions that may consist only of natural language sentences and graphical actions and events (multimodal interactions). The architecture is extended from Lopes [Lop91] and is composed of two decisione centers (heads): an interactional and a reflectional one. The interactional decision center is responsible for the interaction with the user and consists of a handler that activates the adequate low-level executive modules (input interpreters, planners and executive generators of planned actions) that perform the desired actions and use and update the context of interaction. The reflectional module is responsible for detecting faults and deadlocks in a interaction and for propagating solutions that will lead to their repair. It also consists of a handler and of executive modules that use the interactional context as object as well as other knowledge sources. It is shown how this architecture can be applied to multimodal interactions. An implemented new environment for the creation of dynamic lexical databases is described as an application.

Topics: Context sensitive user interfaces, natural language understanding, intelligent multimodal systems.

1 Introduction

Multimodal systems enabling their users to choose among various modes of communication (like spoken or written natural language input, menus and other graphical objects, database query languages, etc.) are increasingly necessary. Their commercial interest follows from the need to allow end users an easy access to computing facilities. However, such systems require powerful architectures for handling multimodal access (text, graphics and sound), as communication mode may change during an interaction, making

[1]This work has been supported by JNICT, INIC, FCT/UNL and Gabinete de Filosofia do Conhecimento.

[2]Owns a scholarship from JNICT, n^o PMCT/BD/1766/91-IA.

context bookkeeping necessary for tracking interaction interruptions, changes of conversational themes, and natural continuation of interrupted interactions. Moreover, they should be able to deal with deadlocks and error situations. In order to achieve these goals, the systems should be aware of the limit of their knowledge and should be able to reason about the current interactional context. According to our point of view, a head (or agent) is needed for conducting the interaction with the user and a second head is needed in order to handle reflection and to help the system to detect and solve unwanted situations such as the following where:

- the user does not behave as expected and does not respond to the system's questions and, as a consequence, the interaction reaches a point where the system's and the user's intentional structures are incompatible,
- there are errors and problems with the executive modules (see section 3), e.g. with
  - the planner (what should be done if the planner cannot build up an effective plan for action? How can a plausible explanation for a faulty input be used for continuing an interaction?),
  - the input interpreter (what can be abducted from an input consisting of errors and incomplete information?),
  - the generator (is it possible to transmit the same information using another mode of communication [FM90]?).

In section 2 we will describe the two-headed architecture defined in order to handle the above mentioned multimodal reflective interactions. The interactional decision center consists of an interactional handler and several low-level executive modules (see section 3). The reflectional-level, handling reflective reasoning (see section 4) is responsible for analyzing the interactional context and solving deadlocks as well as error situations. The architecture is partially implemented using the programming language Prolog and the X-Windows graphical environment (see [Abr89]) and, for the moment, is able to deal only with some error situations.

In section 5 we will describe how this architecture is adapted for the creation of dynamic lexical databases and some examples of a typical session for creating and updating lexical databases will be shown.

Finally, in section 6, some of the problems encountered and requiring additional work will be pointed out, namely the architecture's capability to overcome deadlocks.

2 Architecture

The defined architecture is adapted from the mono-headed two level architecture for intentional participation of natural language interfaces in conversations ([Lop91]).

This architecture consists of an interactional handler whose behavior is described by a deterministic finite-state automaton (see figure 1) implemented by an Interaction Grammar (IG). Besides, there are at least three executive modules (an interpreter of user acts, a planner of the system's future behavior and a generator of planned acts).

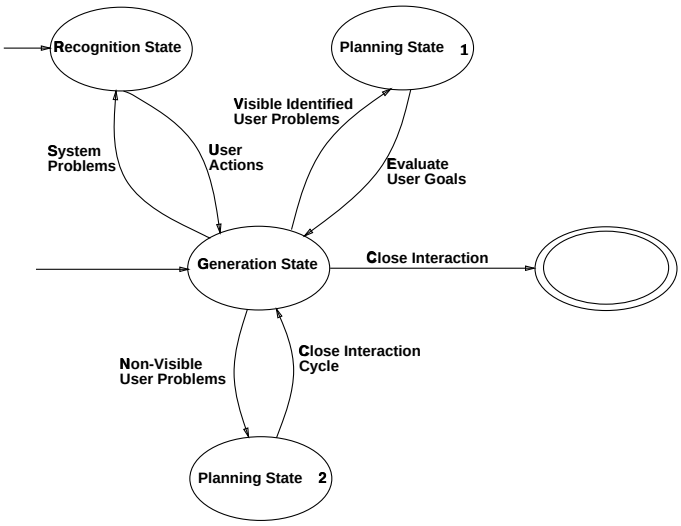

Figure 1: Interaction finite-state automaton

The interactional handler, depending on its current state and on the interaction context, hires the adequate executive module that will perform the task for which it was hired, will update non-deterministically the interactional context and will respond back to the interactional handler allowing it to change deterministically into another state (see figure 1). In the automaton each change of state (edge) has some pre-conditions associated and requires the activation of specific executive modules in order to use and update non-deterministically the interactional context. A deterministic automaton reading a non-deterministic interactional context enables the description of a non-deterministic course of events.

In this paper, a new reflectional-level is defined giving the possibility to incorporate reflective reasoning (see [RVH+91]) in order to reason about the current system's behavior.

The reflectional decision center and the interactional decision center have similar architectures. Both consist of a handler that activates the low-level executive modules in order to use and update the context of interaction. However, their behavior is completely different as they have different executive modules and different long-term goals, namely the interactional level tries to fulfill the user goals and the reflectional level tries to solve the interactional-level problems by analyzing the interactional context. On the other hand, the interactional level interacts with the users, the reflectional level and the knowledge bases while the reflectional level only interacts with the interactional level and the knowledge bases.

In figure 2 a general overview of the system's architecture is given.

The context of the interactional decision center and the context of the reflectional decision center are data structures both defined as 5-tuples *(IG, DCIS, UBS, ARS, KS)* with:

- IG - the Interaction Grammar, defining the finite-state automaton,
- DCIS - the Decision Center Intentional Structure consisting of a triple (DCPG,

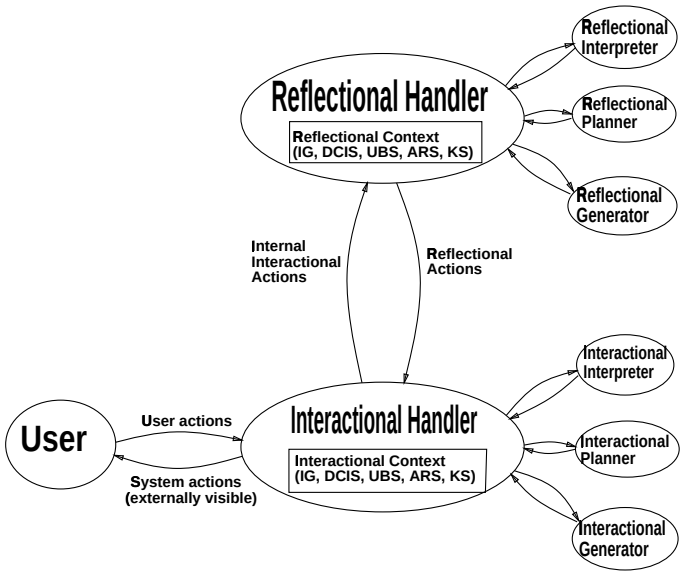

Figure 2: Overview of the architecture

DCAG, DCCG) where:

  - Decision Center Pending Goals (DCPG) is a tree of pending goals and the structure of user acts that induced the goal;
  - Decision Center Activated Goals (DCAG) is a tree of goals, plans for achieving the goals and the user expected behavior;
  - Decision Center Completed Goals (DCCG) is a tree of goals, plans and user behavior;

- UBS - User Behavior Structure consisting of triples (UIA, USA, URA) where:
  - User Identified Acts (UIA) is a tree of user acts and expected system acts;
  - User Selected Acts (USA) is a tree of user acts and expected system acts;
  - User Responded Acts (URA) is a tree of user acts and system acts;
- ARS - Attentional and Rhetorical Structure;
- KS - Knowledge Sources.

The Decision Center Intentional Structure is composed of three structures: the pending goals, the activated goals and the completed goals. These goals are represented as Prolog terms trying to model the decision center intentional structure (see section 3.1.2 for some examples). The Decision Center Completed Goal Structure is a tree which models the decision center goal dependency with each node being composed of the completed goal connected to the list of user acts that induced it, the plan used to achieve the goal and the user behavior corresponding to the head acts. The nodes are inserted into and deleted from the trees by the executive low-level modules, namely the planner and the generator (section 3.1 and 4.1).

The User Behavior Structure is a structure similar to the DCIS trying, however, to describe the user behavior. The user acts are also represented as Prolog terms and are created by the interpreter module. The node structure is composed by the user acts as well as by the expected decision center behavior.

The Attentional and Rhetorical Structure is a DRS (Discourse Representation Structure) of the interaction and, as future work, these discourse structures will be enriched with the respective rhetorical structures (see [KR93, LA91, RL92b, RL92a]).

The Knowledge Sources depend upon the level: in the interactional level the knowledge sources are the lexicon, the user models and the knowledge based information systems and in the reflectional level the knowledge sources include also an image of the interactional level context.

3 Interactional Decision Center

This decision center interacts both, with the user and the reflectional decision center. Other decision centers with the same architecture can be considered for regulating the interaction with specific knowledge sources. It is composed by an interactional handler which activates the executive modules to interpret the input, to plan future actions and to generate system actions. Generated actions can be directed either to the end-user, to the reflectional-level or to other knowledge sources. For example, if a faulty input is detected during an interaction, the reflectional-level is asked to find the fault and to try to repair it. Suppose the fault was caused by an unknown word form and that at the reflectional level a plausible explanation was found, namely that the word is misspelled. Then, new objectives are set back to the interactional level in order to conduct a confirmation dialogue with the user. If the explanation found by the reflectional level is accepted by the user, the reflectional level sends the result of the analyzes to the interactional level and the interaction with the user may continue.

On the other hand, the interactional analyzer receives input from the user, from the reflectional-level and from the consulted knowledge sources. If the reflectional decision center is consulted about a given error, its planner and generator modules produce actions directed to the interactional decision center in order to overcome the problem, as pointed out in the previous example.

Multimodal interactions are supported by the interactional context defined in the previous section. For the purpose of our application (dynamic lexical databases) it was necessary to define the executive modules and the appropriate knowledge sources (i.e. lexicon and users).

3.1 Interactional Executive Modules

The interactional level consists of several low-level executive modules in order to use and update the interactional context.

3.1.1 Interpreter

The user can interact with the system using natural language sentences and performing actions (clicking on a button, selecting an item from a list, etc). On the other hand, the reflectional-level can generate internal messages about actions and send them to the interpreter module. Consulted databases and other information systems may also respond to the interactional level. The actions are then interpreted and transformed by this module into Prolog terms and inserted into the user identified act structures, e.g. *user-clicked-button(+Button)*[3], *user-selected-list(+List), user-entered-text(+Text), meta-goal(+Goal)*[4].

It should be stressed that the interpreter is driven by expectations and, accordingly, it tries to interpret specialized input (meta, user and information system messages).

The analyzer module was implemented in X-Prolog, a Prolog environment enabling the access to the X-Windows functionalities (see [Abr89]).

3.1.2 Planner

This module selects and analyzes the user's acts as well as the reflectional acts and the consulted information system acts that are created by the interpreter module and generates the adequate Prolog terms representing system goals to be solved. These terms are added to the system's pending goals and will be responsible for the future behavior of the system. The generation of goals depends on the actual interactional context. Some examples of possible pending goals are: *ask-user-about(+Goal), inform-user-about(+Goal), ask-meta-about(+Goal), inform-meta-about(+Goal).*

3.1.3 Generator

This module analyzes the system's pending goals and generates the adequate actions. These actions can be of four different types: *actions directed to the user, actions directed to the reflectional level, actions directed to databases and other knowledge sources, new pending goals.*

If the generator chooses to perform interface actions, it can interact directly with the user. These actions can be simple or atomic (e.g. open a window with some help, respond to a question, etc.) or they can be complex natural language sentences.

A pending goal may be too complex to be solved by a simple action and may thus generate a list of new pending goals, for example the goal

- inform-user-about(word(+Lexicon))

may generate two new goals

- access-lexical-database(+Lexicon, -Info), inform-user-about(+Info).

On the other hand, the pending goal may also generate an action to be propagated to the reflectional level like asking to find out an explanation for a faulty input and a solution to overcome the problem.

[3]The sign "+" is used to define output parameters and the sign "-" to define input parameters

[4]In this work the meta level is equivalent to the reflectional level

4 Reflectional Decision Center

This decision center is created to detect and solve situations where the interaction reached a deadlock or where an error occured. It is necessary to enable the system with the capacity of self reasoning as it should be able to reflect and act in order to solve its own problems. The self reasoning capacity can be started when the interactional level asks for help or when the reflectional level detects a deadlock or an error situation (deadlocks will not be covered in this paper).

This center consists of the same structure as the interactional level, i.e. of a handler with a context using modules to execute the desired actions. Thus the architecture is kept simple with the reflectional level being similar to the interactional level consisting of different objects and different executive modules.

The reflectional level interacts with the interactional level receiving internal messages (help messages) and acting in order to try to solve the interactional problems (i.e. creating new goals). In this phase, the reflectional level only acts when it is asked for, i.e. when the interactional level detects an error. In a second phase (future work), this level will have the power to act by itself, analyzing the interactional context and detecting more important problems as deadlocks as well as defining global system strategies such as changing the user model, selecting the adequate mode, etc.

One of the knowledge sources is an image of the interactional context. This means that the reflectional level has a model of the object (the interactional level) and manipulates its object's image. As Reinders et al. ([RVH+91]) point out, this approach has several advantages:

- it is selective - the model is built only with the relevant parts of the interactional context. In our case we have used the interface intentional structure, the user behavior structure and the interaction attentional structure;
- it is specialized - we can define the model with the structure that better fits our aims. We have defined the image with the same structure as the context object.

However, there should be a causal connection between the object and the image (if one changes, the other one should change accordingly). We tried to solve the problem of the integrity image-object making it impossible for the reflectional level to change the interactional context directly. On the other hand, the integrity of the object-image is kept by forcing each context change to be propagated to the image.

4.1 Reflectional Executive Modules

As pointed out, the reflectional level has also several executive modules in order to use and update the reflectional context.

4.1.1 Interpreter

The reflectional level receives input from the interactional level as internal actions (Prolog terms). These actions are simply inserted into the interface identified act structures, e.g. *user-errorfull-input(+Sentence).*

In a second phase (future work), in order to detect deadlocks, the analyzer will periodically generate internal actions in order to analyze the interactional context.

4.1.2 Planner

This module analyzes the reflectional context and the image of the interactional context and generates the adequate Prolog terms. Suppose the interactional level encounters an unparsable sentence. Then, the reflectional planner will build a plan to use the interaction context, the lexical database and the syntactic and semantic information available to infer possible explanations.

4.1.3 Generator

This module analyzes the reflectional pending goals and generates the adequate actions (Prolog goals). It can interact with the lower interactional level and/or with the databases.

In our system the generator can create two kinds of actions: interactional goals *meta-goal(inform-user-about(+Info)), meta-goal(ask-user-about(+Info))* and internal goals *ask-database-about(+Word, -Info), infer-lexical-classification-about(+Word, -Info).*

The interactional actions are sent to the interactional level and will be responsible for the system's behavior trying to overcome the detected problem.

5 An application: Dynamic lexical databases

In this section it will be shown step by step how this architecture is applied to the building of dynamic lexical databases allowing not only the acquisition of lexical data but also the definition of changes in the lexical data model and some examples of the final result will be shown. Two kinds of users are defined: computational lexicographers and end users. Computational lexicographers are allowed to change the interfaces and the lexical data model while end users are only allowed to insert new entries into the database. Moreover, each kind of user has access to specific lexical information. Thus the interactional and reflectional modules were defined and implemented and it was designed the layout of the graphical objects used in the acquisition of the lexicon. Different graphical objects for different kinds of words (verbs, nouns, adjectives, etc.) were created as well as graphical objects for help windows. Finally the user models of lexicographers and end users were defined.

In the implementation of these modules only unknown or misspelled words were taken into account.

In order to solve these problems, the reflectional level tries to abduct the correct information using the lexical databases and the actual context of interaction. If it fails during this phase an order to ask the user for the missing information will be issued. The given information will then be inserted into the database.

An example of a typical working session of a lexicographer during the acquisition of new lexical entries will be shown: Suppose the system is analyzing a natural language

text and the interactional analyzer detects an unparsable sentence. The user identified acts structure will have the term

- unparsable-sentence(+Sentence).

and the planner will insert the following term into the interaction pending goals

- ask-meta-about(fault-repair(unparsable-sentence(+Sentence))).

The detected problem will be communicated to the reflectional level and, taking into account the context and the system's lexical, syntactic and semantic knowledge, it might be possible to conclude that the sentence was unparsable because it contains an unknown word that is probably a verb with a given subcategorization.

A cycle of interaction is then initiated by the interactional level and the lexicographer is asked to confirm the respective information.

By now the tree of the interaction pending goals consists of the term

- ask-user-confirmation-about(+Verb, +Info).

If the user confirms the information, the reflectional level is informed and as a consequence the interaction pending goals is updated with

- change-lexical-database(+Verb, +Info).

The information is inserted into the database and the user identified acts are updated with the result of the previously unparsed sentence.

On the other hand, it is possible to use another system capability, the interface generator, that can be used at runtime in order to change the text and the layout of the interface windows.

6 Conclusions and future work

The designed and implemented architecture for multimodal interactions based on a conversational model showed to have some powerful characteristics: it handles natural language sentences and graphical interactions, it is able to access specific knowledge bases depending on the interactional context, it has the power for reflectively reasoning about errors, it is a flexible interactive machine, enabling the definition of different user models, and it is able to handle interruptions and cycles of interactions.

The application of the above mentioned general architecture to the building of dynamic lexical databases allows the implementation of a lexical system with the power to interact with the user by means of a multimodal interface, allowing the lexicographer to create and to update the lexical databases during an interaction.

Moreover, the proposed two-headed architecture is able to deal with errors and missing information without the need to incorporate a high degree of complexity into the interaction executive modules as the abnormalities are handled on different specialized levels and each executive module is responsible to perform specific actions in a given context. With this modular architecture it is easy to add more specialized features (access to other knowledge bases, interaction using other media) to the system.

However, the internal acts and goal representations are not yet totally defined making necessary a formal knowledge representation for these structures. On the other hand, deadlocks are not yet dealt with, namely the problem of how the reflectional level will detect them and how it will act in order to overcome them. Much of our future work will be certainly concerned about these problems.

To summarize, we have defined an architecture that is able to plan and conduct the system's behavior by means of a multimodal interface including natural language processing capabilities. Moreover, the system has the capacity to handle unknown and non deterministic situations in a very constructive way.

References

[Abr89] Salvador Pinto Abreu. *ALPES X-Prolog Programming Manual.* CRIA/UNINOVA, 1989.

[FM90] Steven Feiner and Kathleen McKeown. Coordinating text and graphics in explanation generation. In *Proceedings of the AAAI'90*, pages 442–449, 1990.

[GS86] B. Grosz and C. Sidner. Attention, intentions and the structure of the discourse. *Computational Linguistics*, 12(3), 1986.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic: An Introduction to Model Theoretic Semantics of Natural Language.* Kluwer, 1993.

[LA91] Alex Lascarides and Nicholas Asher. Discourse relations and defeasible knowledge. In *Proceedings of the 29th Anual Meeting of ACL*, pages 55–62, 1991.

[Lop86] J. G. Lopes. *Conceptualization of an Automatic Interlocutor.* PhD thesis, I.S.T. Technical University of Lisbon, 1986. in Portuguese.

[Lop91] J. G. Lopes. Architecture for intentional participation of natural language interfaces in conversations. In C. Brown and G. Koch, editors, *Natural Language Understanding and Logic Programming III.* North Holland, 1991.

[LS92] J. G. Lopes and A. M. Santos. Portuguese lexicon acquisition interface: Plain. In *Euralex'90 Proceedings.* Bibliograph VOX, 1992.

[RL92a] Irene Rodrigues and José Gabriel Lopes. A framework for text interpretation. In B. Du Boulay and V. Sgurev, editors, *Artificial Intelligence V - Methodology, Systems and Applications.* North Holland, 1992.

[RL92b] Irene Rodrigues and José Gabriel Lopes. Temporal structure of discourse. In *Proceedings of the COLING'92*, 1992.

[RVH+91] M. Reinders, E. Vinkhuyzen, H.Akkermans, J. Balder, B. Bartsch-Spoerl, B. Bredeweg, U. Drouven, F. Harmelen, W. Karbach, Z. Karssen, G. Schreiber, and B. Wielinga. A conceptual modelling framework for knowledge-level reflection. *AIICOM*, 4(2), September 1991.