Robust Semantic Interpretation and Dialog Management in the Context of a CALL Application

Johan Michel
Avenue Laënnec
72085 LE MANS CEDEX 9
johan.michel@univ-lemans.fr

Jérôme Lehuen
Avenue Laënnec
72085 LE MANS CEDEX 9
jerome.lehuen@univ-lemans.fr

Abstract

This paper describes a work about dialog managing in the context of a Computer Assisted Language Learning (CALL) research. In this paper, we choose to focus on the dialog management which is modeled in terms of tasks and methods. The following sections describe the semantic analyzer, the dialog model, and a few results.

1 Introduction

We will present a double model (semantic interpretation and dialogue management) for human-computer dialog in the context of a computer-assisted language learning (CALL) system. In this environment, the learner is implicated in an interaction with a virtual partner around a task (a recipe) to perform in a virtual micro-world (a virtual kitchen) [Lehuen 00], [Michel & Lehuen 02]. Related works exist, we can mention [Hamburger 94]. The following figure shows the user interface of our system:

2 The analyzer

In this specific context, the system will have to deal with incomplete or ungrammatical utterances. So our constraints are the following ones:

- Modularity: it takes place in an existing CALL architecture. More particularly, it is in connection with a virtual environment implemented as a micro-world;
- Robustness: it has to deal with odd utterances. Moreover, even if the learner uses correct words which are not in the lexicon, the interaction has to be continued;
- Non-determinism: it has to be able to produce partial or multiple interpretations for one utterance in order to carry on the interaction. The context of the interaction has to complete or select them.

We start from existing robust methods like "skimming parsing" [Dejong 82] and "chart parsing" [Winograd 83]. But the semantic analyzer we implemented is able to generate lexical hypothesis when unknown words impede its process [Michel & Lehuen 04]. Then, these hypotheses are used to engage a dialogic recovery strategy using the words recognized by the analyzer.

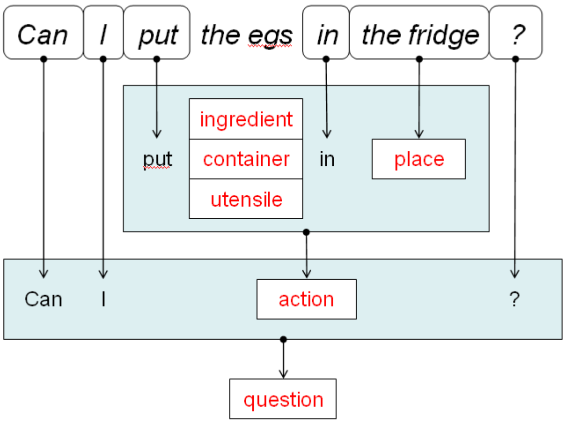


Fig. 1: Example of a syntax-driven hypothetico-deductive analysis

The figure 1 shows how the sentence *"Can I put the egs in the fridge?"* is analyzed as a question on the basis of one (triple) hypothesis about the unknown (and wrong) segment *"the egs"*. In this syntax-driven hypothetico-deductive analysis, *"the egs"* can be an ingredient, a container or a utensil. The analysis has three different steps: the lexical cover checking, the syntactic cover checking and the syntactic recovery. The lexical cover checking verifies if all the words in the utterance belong to the lexicon. The syntactic cover checking verifies if a syntactic pattern can be applied to the utterance. The syntactic recovery reorganizes the utterance to find a syntactic pattern. This recovery succeeds if after having reorganized the sentence, a syntactic pattern is found.

Good lexical cover	good syntactic cover	*« put the eggs in the fridge »* (1)	
	bad syntactic cover	Syntactic recovery succeeds	*« put in the fridge the eggs »* (2)
		Syntactic recovery does not succeed	*« the fridge »* (3)
			« open » (3)
Bad lexical cover	good syntactic Cover	With hypothesis	*« foo put the eggs in the fridge »* (4)
		Without Hypothesis	*« put the foo in the fridge »* (5)
	bad syntactic cover	Syntactic recovery Succeeds	*« put in the frige foo the eggs »* (6)
		Syntactic recovery does not succeed	*« foo the eggs foo »* (7)
			« foo » (8)

Table 1. Examples of cases the analyser must handle

Different interaction strategies can be chosen given the analysis results. The first step is checking if the analysis's results correspond to the applicative (state of the applicative task) and interactive context, in this case, the analysis is validated. The second step is creating the partner's reaction from the applicative and interactive context.

3 The dialog model

The second part of our work focus on the knowledge of the dialog and the domain levels which are both modeled as tasks and methods. This approach is coming from research on generic mechanisms for problem-solving. It enables to rationalize

the behavior of the system and provides a framework to design an abstract, implementation-independent description of problem-solving process (fig. 2). In our case, the domain level is only a pretext to engage dialog situations and to make rise linguistic problems. So, the dialog level is weakly connected with the domain level: the repair strategies are more about language and less about the task going on.
Figure 3 contains some tasks and methods to engage a dialog. You can see four tasks, one decomposition method, one iteration method and two execution methods.

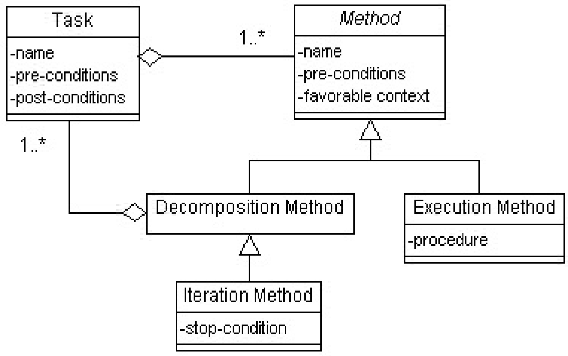


Fig.2: Task-Method framework as an UML class-diagram

.

```
(task (name     T-structure-dialog)
      (methods  M-structure-dialog))

(method-decomp (name   M-structure-dialog)
               (tasks  T-open-dialog T-dialog T-close-dialog))
(task (name     T-open-dialog)
      (methods  M-open-dialog))

(method-exec (name      M-open-dialog)
             (function  F-open-dialog))

(deffunction MAIN::F-open-dialog ()
   (assert (to-write "Hello, can you explain to me how to make a
                     chocolate cake?"))
(task (name     T-dialog)
      (methods  M-dialog))

(method-iter (name       M-dialog)
             (stop-cond  "(stop dialog)")
             (tasks      T-listen T-respond))
(task (name     T-listen)
      (methods  M-listen))

(method-exec (name      M-listen)
             (function  F-listen))

(deffunction MAIN::F-listen ()
   (assert (waiting-learner-utterance))
   (focus IHM))
```

Fig. 3: Some tasks and methods to engage a dialog

According to [Trichet & Tchnounikine 99], a Task-Method model can be decomposed into four layers, each layer manipulating data from the lower one. The highest layer, named control, deals with the way the modelized task is performed. It can be a simple algorithm, a classical expert system, or even a higher-level Task-Method engine (fig. 4). In our case, the control of the dialog level is performed by a simple independent algorithm, whereas the control of the domain level is performed by specific methods of the dialog level.

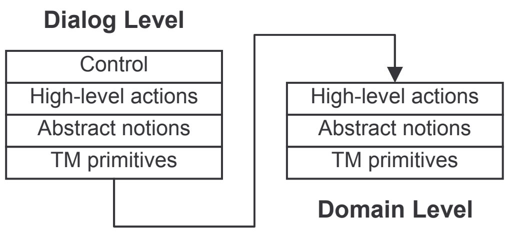


Fig.4: Task-Method model based on 4 layers

These models are fully implemented in Jess, a rule-based environment for the Java platforms. A first experiment has been conducted to test the system and its robustness. In the interaction logs, we have observed the different awaited dialog situations.

4 Examples of dialogs

In the logs, we found the different interaction cases we expected. For a given group, on forty-six exchanges, we counted thirty-two exchanges linked to the task without any problems (Figure 5), fourteen exchanges with linguistic problems (Figure 6), and one exchange with a problem about the task (Figure 7). The exchanges are translated from French.

```
Learner> Take the milk
Partner> I take the milk from the fridge.
Learner> Take the cream
Partner> I take the cream from the fridge.
```

Fig. 5: Exchanges without problems

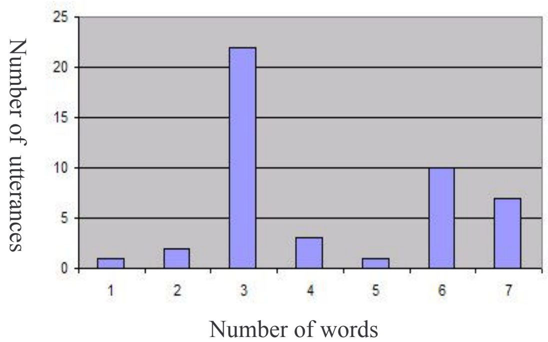
```
Learner> Open the cupboard at the top on
the righ
Partner> I do not understand when you write
« Open the cupboard at the top on the
righ». Which cupboard do you want to open?
Learner> Open the cupboard at the top on
the right
Partner> I open the cupboard.
```

Fig.6: Exchanges with linguistic problems

```
Learner> Take the milk
Partner> I understand, but it has already
been done
```

Fig.7: Exchange with a problem about the task

The learner's utterances were relatively short (from one to seven words) but this is mainly due to the nature of the task. The figure below illustrates this phenomenon for a given dyad:



References

[Dejong 82] G. Dejong 1982, *An overview of the FRUMP system*, In: "Strategies for natural language processing", Lehnert & Ringle, Erlbaum, Hillsdale, pp. 149-176.

[Delorme & Lehuen 03] F. Delorme and J. Lehuen 2003, *Dialog Planning and Domain Knowledge Modeled in terms of Tasks and Methods - A Flexible Framework for Dialog Managing*, In: ISMIS'03, Maebashi City (Japan), Springer-Verlag "Foundations of Intelligent Systems", Lecture Notes in Computer Science, pp. 689-693.

[Hamburger 94] H. Hamburger 1994, *Foreign language immersion: Science, practice, and a System*, In: JAIED Special Issue on Language Learning, vol. 5(4) pp. 429-453.

[Lehuen 00] J. Lehuen 2000, *A Dialogue-Based Architecture for Computer Aided Language Learning*, In: AAAI Fall Symposium on "Building Dialog Systems for Tutorial Applications", pp. 20-27.

[Lehuen & Luzzati 99] J. Lehuen and D. Luzzati 1999, *Acquisition coopérative d'une compétence langagière interprétative en dialogue homme-machine*, In: TALN'99, Cargèse (Corse), pp. 357-362.

[Michel & Lehuen 04] J. Michel and J. Lehuen 2004, *Un analyseur hypothético-déductif non déterministe pour l'apprentissage et la pratique d'une langue*, In: TAL & Apprentissage des Langues, Grenoble (France), pp. 13-22.

[Michel & Lehuen 02] J. Michel and J. Lehuen 2002, *Conception of a Language Learning Environment based on the Communicative and Actional Approaches*, In: ITS'2002, Biarritz (France), pp. 651-660.

[Trichet & Tchounikine 99] F. Trichet and P. Tchounikine 1999, *DSTM: a Framework to operationalise and Refine a Problem-Solving Method modeled in terms of Tasks and Methods*, In: International Journal of Expert Systems with Applications, vol. 16, pp.105-120.

[Winograd 83] T. Winograd 1983, *Language as a Cognitive Process*, Syntax, Addison-Wesley Publishing Compagny.