STARTING TO COOK A TUTORING DIALOGUE SYSTEM

Filipe Martins, Joana Paulo Pardal, Luís Franqueira, Pedro Arez and Nuno J. Mamede

Spoken Language Systems Laboratory, L^2F – INESC-ID
IST / Technical University of Lisbon
R. Alves Redol, 9 – 1000-029 Lisboa, Portugal

ABSTRACT

This paper presents a system that helps you cook a recipe through a spoken dialogue tutoring session. We report our experience while creating the first version of a tutoring dialogue system that helps the user cook a selected dish. Having a working framework to support us with the creation of the cooking assistant, the main challenge we faced was the change of paradigm: instead of the system being driven by the user, the user is instructed by the system. The result is a system capable of dictating generic contents to the user. On top of it, the system can be used in several domains where the goal is not the replacement of the user but providing some assistance while (s)he performs some procedural task.

Index Terms— Natural Language Interfaces, Speech Communication, Home Automation, Natural Language, Data Models

1. INTRODUCTION

Cooking is something that everybody ends up doing sometime in his/her life. Some love it, some hate it. Some have been cooking for a lifetime, some are just beginning. No matter what expertise level the user has, most of the time her/his hands are busy and, on top of it, dirty. In such a common scenario, manually handling a recipe book is something to avoid, not only to keep the book tidy, but also to make the task easier and safer. A system that helps the user by dictating the needed steps while his hands and eyes are occupied with the cooking tasks is much desired.

Being so, our goal is to develop a spoken dialogue system that provides assistance in reading the procedure and detailing all steps that are unclear to the user by lack of expertise. Based in our own cooking experience, we built a prototype system that helps the user while (s)he performs some task, and we experimented this with the cooking domain. Also, as not everyone needs the same kind of help while cooking, we allow the system to adapt itself to the users' needs and expertise. The starting point of our worked was DIGA [1], an existing framework that allows the creation of spoken dialogue systems. This paper focuses on the changes required to create a new cooking assistant.

As in the Intelligent Procedure Assistant (IPA) system [2] this system was designed considering that the user has his/her hands busy while executing the task being tutored. IPA is a multi-modal spoken dialogue system aimed at providing guidance and support to astronauts on the International Space Station (ISS) during the execution of large procedural tasks. In our system, the tasks are smaller, and the environment is different. However, our users have different levels of expertise (while astronauts are expected to be skilled in the tasks they perform at the ISS). Because of this, tasks need to be detailed as much as possible. To allow this feature, tasks are described hierarchically and we try to reuse as much as possible the steps common across different tasks. Reading a recipe tree-like description left-to-right we have the main tasks. Going top-down we have the steps need for each task.

As pointed out by Wasinger [3], when compared to a paper cookbook, a system like this has advantages such as being easier to use because hands and eyes are free from the book. Wasinger proposed the development of a home cooking assistant to find recipes, detail recipes and allow for the addition, removal and modification of recipes. However it is not clear how the system guides the user through the cooking process.

This paper follows with a brief description of the existing framework in section 2; details of the recipe used to cook up the dialogue system are presented on section 3, where the interaction and the recipe models are described. A semi-automatic natural language based tool was built to enlarge the set of recipes (subsection 3.4). To test the domain independence of the solution, a second tutoring system was built: a car-assistance tutor described on section 4. The paper closes with some conclusions and future work directions.

2. BACKGROUND

Most dialogues systems are created to help the user access some back-end system. Usually the system construction is driven by a set of tasks, and its main responsibility is to collect the information required to execute some external service. The starting point of DIGA, the framework we used to built our system, is no different.

DIGA, DIaloG Assistant, is a domain-independent framework for spoken dialogue systems [1] that was initially built

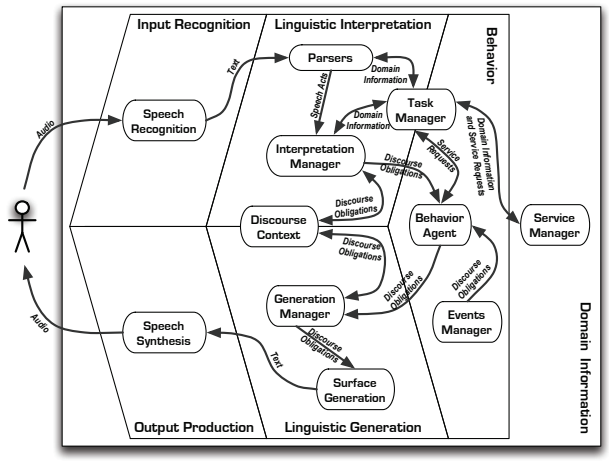

Fig. 1. DIGA's architecture.

as the basis of two distinct applications: a butler that controls an home intelligent environment; and an interface to remotely access information databases (like bus timetables).

DIGA has three main modules: an Input-Output Manager (that controls an Automatic Speech Recognition module [4], a Text To Speech module [5] and a Virtual Agent Face [6]); STAR, a Dialogue Manager (that interprets the user intentions and generates output messages [7]); and a Service Manager (that provides a dialogue manager interface to execute the requested services, and an external application interface [8]).

The architecture of STAR is shown on Figure 1. It is frame-based: every domain is described by a frame, composed by domain slots that are filled with user requests until a service can be executed [9]. Recently, when creating two new telephone-based services (home banking and a personal assistant) DIGA's team faced several engineering challenges that lead to significant changes in the previous architecture [10, 11].

3. THE COOK TUTOR

To create the dialogue system a conceptual model that represents the recipes was needed. The chosen representation needs enough expressiveness to: allow the selection of a recipe; allow different levels of detail according to the user expertise. Also, the interface design needs attention in order to provide all the useful means of manipulation of the recipe like the level of detail, or information about the current step.

3.1. Interaction design

When mimicking the wanted system behavior, it became clear that the main instructions the user could give to the system would be: move one step forward; move one step backward; detail the current step; return to the previous level of detail; end the tutoring task. These instructions were encoded on primitive services: *next*, *previous*, that gave the user control over the dialogue flow. A mechanism to enter into a more detailed level of explanation is provided with *how* ("How can I do that?"). To return to the previous level the primitive is *return* ("OK."). Also, during some initial tests, it came to our attention that it was also useful to provide a mean of asking the system to repeat the last spoken instruction. This led to another primitive: *what* ("What did you say?").

To minimize the needed changes on the framework, these primitives were implemented as services in the cooking tutoring domain. Each service has its description and rules. Being so, an interaction could be as the one shown on Figure 2[1]:

User: Chef, I want to cook a chocolate cake.
System: OK! You will need the following ingredients.
User: Next
System: 4 oz of sweet or semisweet dark chocolate.
User: Next
System: 5 eggs
User: What?
System: 5 eggs
`(...)`
System: Separate egg whites and egg yolks
User: Next step
System: With an electric mixer whisk the eggs whites until forming stiff peaks
User: How?
`(down a recipe level)`
System: Put eggs whites in a container
User: How?
System: Let me show you with a brief movie...
`(final recipe level - plays related video)`
User: OK! Got it! Next step
`(end of this level - go up one level)`
System: Add sugar as needed
`(...)`
System: And that's it! Enjoy!

Fig. 2. An interaction example.

3.2. Interface Design

Because we want the user to better understand what is being explained, the interface design is also important. Some changes were made to the existing interface to include a window that displays images and video to provide additional visual information. Figure 3 shows a screenshot of the system while showing a video on how to whisk egg whites.

[1] Although our system works with Portuguese language, we used English in the examples in order to allow a broader understanding of the paper.

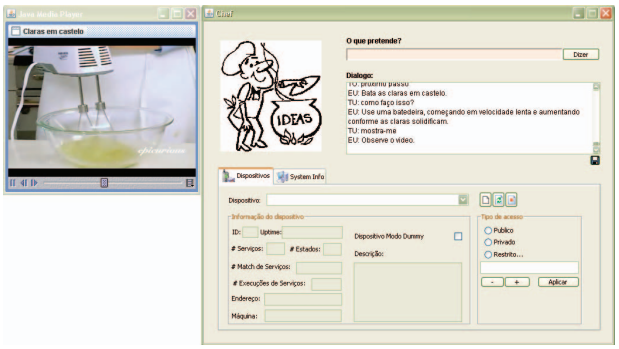

Fig. 3. Screenshot of the Chef's interface.

3.3. Modeling Recipes

We created a proper model for a *recipe* where they are described as a list of ingredients, and an ordered list of steps, where each step can be detailed into a new simpler recipe (according to each level of detail and expertise). According to this model we defined a set of recipes in a XML file. Figure 4 shows excerpts of a chocolate cake recipe.

```
<dictate>
  <step>
    <ingredient>
      4 oz of sweet dark chocolate
    </ingredient>
  </step>
  (...)
  <step>
      separate egg whites from egg yolks
  </step>
  <step>
      whisk the eggs whites
      until forming stiff peaks
    <sub>
      <step>
          put eggs whites in a container
      </step>
      <step>
          beat the mixture vigorously
        <sub>
          <step>
            <video path="./recipes/mix_eggs"/>
          </step>
        </sub>
      </step>
    </sub>
  </step>
  (...)
</dictate>
```

Fig. 4. Example of a (partial) recipe.

In each recipe we have a set of initial steps that represents the list of ingredients needed to perform it. This is marked with a nested node with an ingredient. Then we have a sequence of main steps, each of which can be detailed into simpler steps in a hierarchical structure. When no further spoken explanation is possible, at some point, multimedia files can be used to help with the difficult task. This is done by showing images or videos. To play the multimedia files (both images and video) the framework had to be extended.

The nested model and the correspondent primitives allow the system to provide the adequate level of detail when describing the main task as needed and related to each step. This allows the system to be easily used by both an experienced mother that cooks every day; and by a young boy that is learning to cook his first cake.

A more comprehensive model of recipes is available in an ontology [12] on the cooking domain. The model we use here is a simplification that provided enough elements to test if such a system could be built over the DIGA framework.

3.4. Acquiring recipes

When the prototype was finished we agreed that adding recipes by hand-coding would be too much of a burden so we decided to look for other options. Given the huge amount of recipes that can be found in books and dedicated web sites in the internet we decided to create a database that could be automatically populated with recipes.

However, the recipes that we found are for human eyes. To create a machine readable version of those recipes a linguistic-based tool was used [13]. This tool converts a recipe into a sequence of SQL commands that can be executed to fill a database. This also takes into consideration the concepts' definition on the cooking ontology, OntoChef [12]. Currently, this tool is run offline to enrich it.

4. CAR ASSISTANCE TUTOR

To test our solution, and its domain-independence, a second prototype was developed: a tutor for car assistance. The goal of this second dialogue system is to help the user while performing tasks related with car maintenance like changing a tire or checking the engine oil-level. Spoken language is very useful in these scenarios since we need both hands free to perform the intended tasks. Again, going through the required manual with dirty hands is an option to avoid.

The procedures are described in XML files with the same structure and the available interactions are the same. The procedures made available were also hand-coded. The model defined for the recipes is generic and allows the system, in both prototypes, to instruct the user providing different amounts of information according to user needs and expertise.

5. CONCLUSIONS AND FINAL REMARKS

We presented a tutor dialogue system built over the DIGA framework that is designed to engage in task-oriented dialogues in one of several task domains. Modeling the types of interaction needed as services in the dialogue system eased

the creation process, since only minimal architectural changes were needed to implement this new system.

Work is still needed, but the prototype is up and running. In this version, four recipes were hand-coded with up to four levels of nested steps. The car assistant tutor provides help when changing a tire. A little multimedia tool was created to allow the system to play videos and show images during the tutoring session. This tool might be incorporated in the framework to enrich it.

The current version of the system takes hand-coded XML file recipes. Current work includes the use of an existing cooking ontology, as suggested by Paulo Pardal [14] incorporating the recipes acquired by an automatic tool (previously described on Section 3.4). This automatic tool was tested with 8 Portuguese recipes and achieved 80%–90% of precision and 67%–88% of recall.

The selection of the recipes is still an issue as no indexing of the recipes is available yet. By using OntoChef that problem can be easily overcome since the recipes can be organized by several criteria. Also, the use of the extraction tool to import recipes on-demand can be explored.

The use of OntoChef delivers another interesting result: the system could reason about the tasks and plans. For instance, by understanding when the user mention the output of some intermediate task, like in the following example:

System: Separate egg whites and egg yolks.

User: What do I do with the whites?

The system needs to know that after the task the user has "egg whites" and "egg yolks" as a result.

As for the multimedia files, in the future, the framework could be adapted to open web pages. That would allow us to take advantage of information available on the internet or previously recorded cooking sessions annotated according to OntoChef, similar to what is done in the Smart Kitchen project [15].

Related to health care, it would be interesting to allow the selection of recipes according to any diet constraints.

On the interface design there are also some enhancements planned, like redesigning the main window in order to make it clearer for the user. We are also considering using videos from free available cooking podcasts (like `startcooking.com`, Wikipedia or YouTube) and use improved images to help on the selection of the ingredients and their measures.

Allowing different languages like Spanish or English by translating the cooking ontology is also in our plans.

Acknowledgments

This work was funded by PRIME National Project TECNOVOZ number `03/165`. It was also partially funded by DIGA, project `POSI/PLP/14319/2001` of Fundação para a Ciência e Tecnologia (FCT). Joana Paulo Pardal is supported by a PhD fellowship from FCT (`SFRH/BD/30791/2006`).

6. REFERENCES

[1] João Paulo Neto, Nuno J. Mamede, Renato Cassaca, and Luís C. Oliveira, "The development of a multi-purpose spoken dialogue system," in *EUROSPEECH*, 2003.

[2] G. Aist, J. Dowding, B. A. Hockey, M. Rayner, J. Hieronymus, D. Bohus, B. Boven, N. Blaylock, E. Campana, S. Early, G. Gorrell, and S. Phan, "Talking through procedures: An intelligent space station procedure assistant," in *EACL Software Demos*, 2003.

[3] Rainer M. Wasinger, "Dialog based user interfaces featuring a home cooking assistant," 2001.

[4] Hugo Meinedo, *Audio Pre-processing and Speech Recognition for Broadcast News*, Ph.D. thesis, Instituto Superior Técnico (IST), Universidade Técnica de Lisboa (UTL), 2008.

[5] S. Paulo, L. Oliveira, C. Mendes, L. Figueira, R. Cassaca, C. Viana, and H. Moniz, "DIXI - A Generic Text-to-Speech System for European Portuguese," in *PROPOR*. 2008, LNCS, Springer.

[6] Márcio Viveiros, "Cara falante – uma interface visual para um sistema de diálogo falado," 2004, IST, UTL, Graduation Thesis.

[7] Márcio Mourão, "Gestão e representação de domínios em sistemas de diálogo," M.S. thesis, IST, UTL, 2005.

[8] Renato Cassaca and Rui Maia, "Assistente electrónica," 2002, IST, UTL, Graduation Thesis.

[9] M. Mourão, R. Cassaca, and N. Mamede, "An independent domain dialogue system through a service manager," in *EsTAL*. 2004, vol. 3230 of *LNCS*, Springer.

[10] F. Martins, A. Mendes, M. Viveiros, J. Paulo Pardal, P. Arez, N. Mamede, and J. Neto, "Reengineering a domain-independent framework for spoken dialogue systems," in *SETQA for NLP (ACL workshop)*, 2008.

[11] F. Martins, A. Mendes, J. Paulo Pardal, N. Mamede, and J. Neto, "Using system expectations to manage user interactions," in *PROPOR*. 2008, LNCS, Springer.

[12] Ricardo Ribeiro, Fernando Batista, Joana Paulo Pardal, Nuno J. Mamede, and H. Sofia Pinto, "Cooking an ontology," in *AIMSA*. 2006, vol. 4183 of *LNCS*, Springer.

[13] Telmo Ramos Machado, "Extracção de informação – introdução automática de receitas de acordo com ontologia," M.S. thesis, IST, UTL, 2007.

[14] Joana Paulo Pardal, "Dynamic use of ontologies in dialogue systems," in *Proc. NAACL-HLT 2007 DC*, 2007.

[15] Michael Schneider, "The semantic cookbook: Sharing cooking experiences in the smart kitchen," in *IET*, 2007.