DJ GoDiS: Multimodal Menu-based Dialogue in an Asynchronous Information State Update System

David Hjelm, Ann-Charlotte Forslund, Staffan Larsson, Andreas Wallentin
GU Dialogue Systems Lab
Göteborg University, SE 405 30 Göteborg, Sweden
{sl,forslund,dhjelm,andreas}@ling.gu.se

Abstract

We present a demo where menu-based spoken dialogue is used simultaneously with a graphical user interface menu system as an interface to a mp3 player. Input is accepted in either modality, while output is presented in both modalities simultaneously. The system is implemented using a new asynchronous version of TrindiKit. For modality fusion and multimodal generation we are using multimodal GF grammars.

1 Introduction

In this demo we present work on multimodal menu-based dialogue currently being carried out at the Göteborg University Dialogue Systems Lab as part of the TALK project[1]. We are adding support for multimodality to GoDiS and TrindiKit, as well as making use of recent simplifications and improvements on TrindiKit. For modality fusion and multimodal generation we are using multimodal GF grammars.

The basic idea is to use menu-based spoken dialogue (Larsson et al., 2001) together simultaneously with a graphical user interface menu system. Input is accepted in either modality. Output is presented in either or both modalities (spoken interaction and GUI interaction) depending on whether each modality currently provides an information channel between system and user. We believe this is a simple yet very useful way of exploring the benefits of multimodal dialogue. In addition, it has the advantage of subsuming and extending the already familiar menu-based-GUI-style interface.

As a showcase we have implemented the DJ GoDiS application, a multimodal interface to a mp3-player.

2 GoDiS, TrindiKit and GF

GoDiS (Larsson, 2002) is an experimental dialogue system implementing a theory of Issue-Based Dialogue Management based on Ginzburg's concept of Questions Under Discussion (QUD). GoDiS is implemented using the TrindiKit, a toolkit for implementing dialogue move engines and dialogue systems based on the Information State approach (Traum and Larsson, 2003). GoDiS has been adapted to several different dialogue types, domains, and languages, including menu-based action-oriented dialogue when acting as an interface to a mobile phone (Larsson et al., 2001) or VCR. To enable multimodal interpretation and generation, we have

[1]Talk And Look, Tools for Ambient Linguistic Knowledge, EC Project IST-507802

recently integrated the Grammatical Framework (Ranta, 2004) into TrindiKit and GoDiS, using the Open Agent Architecture (OAA). GF is a grammar formalism based on type theory. The division between abstract and concrete syntax enables grammars to be written in parallel for different languages, sharing the same abstract syntax.

3 Asynchronicity in TrindiKit4

In TALK, all partners have agreed to use OAA as a common interface. Previous versions of TrindiKit were compatible with OAA but required dialogue systems to be run by a TrindiKit agent which would then call other agents when necessary. No solvables were offered by TrindiKit to the OAA agent community.

TrindiKit4 offers the possibility to distribute system components across several OAA agents. As a consequence, a number of input and output modules can run simultaneously, and update and inspect the TrindiKit Total Information State (TIS) independently. For instance, a module can listen continuously for speech and update the TIS only when speech has been detected without blocking the rest of the system. The capabilities of each TrindiKit component are published as solvables to the OAA community.

System coordination is done by a special purpose control module, which can be set up to monitor certain TIS variables and execute an associated control algorithm when they are set to a certain value. This can be used e.g. to notify the interpretation module that the input module has updated the TIS. The control algorithm can contain calls to TrindiKit modules or OAA agents as well as TIS updates and checks. Any number of control algorithms can be run in parallel. As OAA enables asynchronous processing, a previous implementation of asynchronicity in TrindiKit has been removed.

In DJ GoDiS, this architecture is exploited by having multiple input and output modules to allow for spoken and GUI-based (as well as written) communication in parallel. All input modules write to the same TIS variable, an input queue. After a certain amount of user inactivity, the contents of the input queue is copied into a string input variable and is considered to make up one user turn. The setting of this variable triggers interpretation, system update, dialogue move selection, generation and system output.

4 Menu-based multimodal dialogue

This approach offers what we believe to be a very flexible and intuitive multimodal interface to any device that can be operated using a standard menu-based GUI interface. Several modes of interaction emerge from a simple setup, including the following:

- The user may use the menu-based GUI in the normal way, without bothering with speech
- The user may use the spoken interface without bothering with the GUI
- The user may make menu choices using speech; these will have the same effect on-screen as making the menu choice by pointing and clicking
- The user may exploit GoDiS' flexible dialogue management to bypass the menu system and give commands and/or provide information as (s)he sees fit. Again, any spoken interactions will result in the corresponding menu options appearing on-screen, thus enabling the user to freely switch modality or combine modalities as desired at any point in the interaction.
- No user-modeling is done apart from keeping track of the shared dialogue state; user adaptation emerges instantly

```
playTask : Task;
volumeTask : Task;

play : Action playTask;
raise_volume : Action volumeTask;

madonna : Object playTask;
level : Object volumeTask;

makeRequest : (t : Task) -> Action t ->  Request t;
makeAnswer : (t : Task) -> Object t -> Answer t;

makePair : (t : Task) -> Action t -> Object t -> Pair;
```

Figure 1: Dependent types in the GF grammar for DJ GoDiS. The dependent type `Task` is used to make sure that actions and objects within the same utterance belong to the same task.

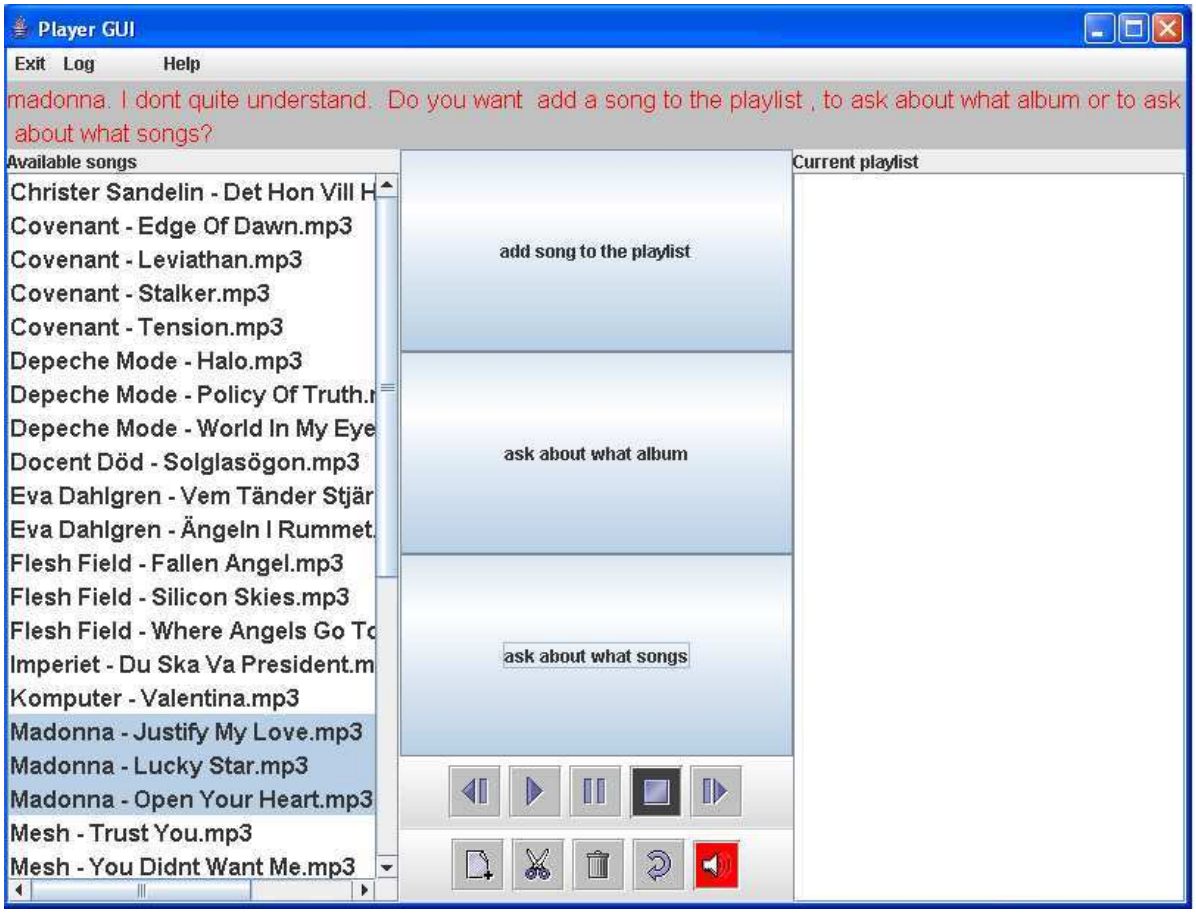

Figure 2: DJ GoDiS GUI. The user has just said "Madonna".

as an effect of the user's choice of modality from utterance to utterance

5 Multimodal grammars in GF and GoDiS

We distinguish between integrated multimodality, where modalities are combined in the same concrete syntax, and parallel multimodality, where each modality has its own concrete syntax (Bringert et al., 2005). In GoDiS we use an integrated multimodal input grammar to parse GUI and speech input and two parallel output grammars generating GUI and spoken output respectively.

The abstract syntax of the two grammars uses dependent types to pose constraints on dialogue move sequences. An excerpt from the abstract syntax is given in figure 1.

GF also support the generation of context free speech recognition grammars. We use this facility to generate a Nuance speech generation grammar from the natural language part of the input grammar.

6 DJ GoDiS functionality

The DJ GoDiS system is capable of performing standard mp3 player tasks, such as playing songs, creating playlists and controlling the volume. The user can also pose queries about e.g. what songs, artists and radio stations are available.

The following example dialogue shows how DJ GoDiS tries to figure out the user's goal by posing a clarification question. (This process is described in Larsson (2002) as "dependent accommodation"):

usr> (CLICKS ON A SONG IN PLAYLIST)
sys> `Do you want to play or remove from playlist?`
(A MENU APPEARS WITH TWO BUTTONS: [PLAY] AND [REMOVE])
usr> `Play`
sys> `OK, play.`
([PLAY] BUTTON IS HIGHLIGHTED)

The next example shows how the user can use two modalities at once to perform a task:

usr> `What songs are there by Madonna?`
sys> (DISPLAYS ALL SONGS BY MADONNA)
usr> `Play this one`
(CLICKS ON SONG 'LUCKY STAR')
sys> `OK, play`
(SONG IS ADDED TO PLAYLIST)
([PLAY] BUTTON IS HIGHLIGHTED)

Figure 2 shows the DJ GoDiS GUI. The user has just said "Madonna". All songs by Madonna are highlighted and three buttons are shown, representing menu choices: [ADD SONG TO PLAYLIST], [ASK ABOUT WHAT ALBUM] and [ASK ABOUT WHAT SONGS].

Acknowledgements

The research reported here was funded by TALK (Talk And Look, Tools for Ambient Linguistic Knowledge), EC Project IST-507802.

References

Björn Bringert, Robin Cooper, Peter Ljunglöf and Aarne Ranta. 2005. Development of multimodal and multilingual grammars: viability and motivation. Deliverable D1.2a, TALK.

Staffan Larsson, Robin Cooper, and Stina Ericsson. 2001. menu2dialog. In *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 41–45.

Staffan Larsson. 2002. *Issue-based Dialogue Management*. Ph.D. thesis, Göteborg University.

Aarne Ranta. 2004. Grammatical framework. a type-theoretical grammar formalism. *Journal of Functional Programming, vol. 14:2. 2004*, 14(2).

David Traum and Staffan Larsson. 2003. The information state approach to dialogue management. In Ronnie Smith and Jan Kuppevelt, editors, *Current and New Directions in Discourse & Dialogue*. Kluwer Academic Publishers.