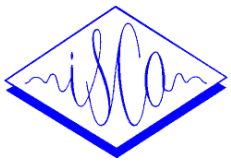

3rd International Conference on
Spoken Language Processing (ICSLP 94)
Yokohama, Japan
September 18-22, 1994

RAPID PROTOTYPING OF A DIALOGUE SYSTEM USING A GENERIC DIALOGUE DEVELOPMENT PLATFORM

Lars Bo Larsen, Anders Baekgaard

**Center for PersonKommunikation, Institute of Electronic Systems,
Aalborg University, DK-9220 Aalborg Denmark.
Email: lbl@cpk.auc.dk, ab@cpk.auc.dk**

1. Abstract

This paper presents the development of a simple, well-defined dialogue system, utilising a Generic Dialogue System Platform for prototyping. The application domain of the present system is an automated book club service.

The man focus of the paper is the description of the dialogue design and implementation processes, together with a description of the underlying principles and architecture of the platform.

The dialogue was designed, and represented as a set of dialogue graphs. A close correspondence between the graphs and the DDL (Dialogue Description Language) implementation is demonstrated, being one of the main reasons for the short implementation phase.

2. Introduction

The work reported in the present paper is concerned with the development of a Spoken Dialogue System.

Automatic Voice-Response (VR) systems for e.g. information retrieval has spread rapidly for a number of years, and commercial speech-activated systems are now beginning to appear. Consequently, it is relevant to investigate the possibilities of transferring of existing VR-applications into speech-activated systems. Therefore, we have chosen to implement the task of an existing VR-application as a Spoken Dialogue System.

A Generic Dialogue System Platform (GDP) has been developed at the Center for PersonKommunikation (CPK) (see [1], [2], [3]). The GDP has been used for a number of applications (see [4], [5], [6], [7]), but we wanted to investigate the potential of the GDP a an application generator

In the following sections a distinction is made between the ***service***, by which is understood the final dialogue system, and the ***demonstrator***, which is a prototype of the final system. This distinction is important, as the main topic of the present paper is the design through prototyping of dialogue systems.

The generic dialogue development platform is described in section 3. The emphasis is put on the dialogue design and implementation process. This is reported in section 4, together with a description of the system functionality. The experiences so far are discussed in section 5.

3. The Generic Dialogue System Platform

The demonstrator is implemented using a generic dialogue system platform. The platform consists of a number of modules that constitute the core of a dialogue system (see Figure 1), i.e. a

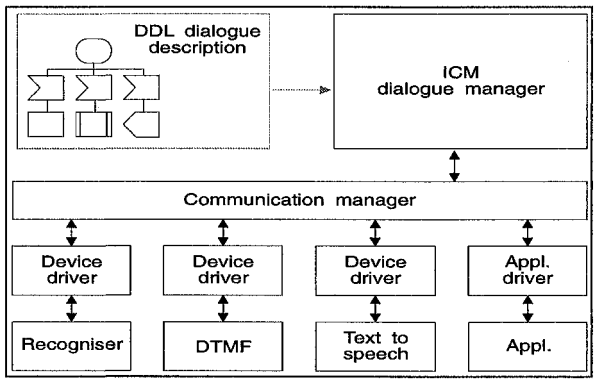

Figure 1. Architecture for Generic Dialogue System Platform

dialogue manager (ICM), a communication system, a well-defined protocol for interfacing to input/output devices and applications, and a graphical tool for describing dialogues. The tool supports the dedicated language DDL - Dialogue Description Language [1], [3].

DDL is a compound language consisting of three levels: a graphical level based on SDL [8] for describing the dialogue structure and the control structure, a frame level for describing data structures such as lexica, and textual level for implementing computations etc. as in traditional textual based programming languages. The three levels are used in two ways: 1) the frame and tex-

tual levels each are used for declaring data structures, functions etc. with a global scope, and 2) each SDL symbol occurring in a diagram (see Figure 3.) has frame and textual levels attachments that in detail defines the meaning of the symbol.

The DDL Tool supports the developments of dialogue descriptions in DDL. It has the necessary drawing facilities and provides a a number of features such as instant syntax check, interactive debugging of the demonstrator etc.

The ICM dialogue manager controls the dialogue on the basis of the description provided by the DDL Tool. It interacts with the speech recogniser in a way which ensures that strong constraints are applied during the speech recognition process. The constraints are determined dynamically during the course of the dialogue. This results in improved reliability for the speech recogniser, and increased performance in terms of speed and overall size of grammar/vocabularies for an application. For further details, refer to [3].

The speech recogniser currently used within the GDP is based on HMMs of whole- or subword units. Spotting of words and phrases is supported. The recogniser runs in real time with a vocabulary up to 100 words, and interfaces directly to the telephone net [9].

Furthermore, the GDP contains a Natural Language parser which transforms the recognised string into a semantic representation [10].

In the present case, the application database for the demonstrator was implemented directly in DDL for convenience.

4. The Demonstrator

This section starts with an overall description of the application domain. The dialogue design process and the corresponding DDL-implementation are then recounted in detail.

4.1 Application Domain

As mentioned in Section 2, the application domain has been chosen to be that of an "Automated Book Club Service". It is representative for a broad class of *automated telephone based ordering systems*. This class ranges widely from e.g. systems for handling private small-ads for a newspaper to home-banking systems. The systems are characterised by (at least for the present) a simple, machine controlled dialogue structure often organised as a hierarchy of menus. As it can be very tedious for the experienced user to go through a large number of menus, short-cuts are often incorporated into the dialogue structure. For the same reason most systems allows "cut-thru" by the user.

A common characteristic of book clubs is that the members once in a month have to decide whether he/she accepts an offer of "the book of the month". The member has to explicitly reject the offer, otherwise the book will automatically be mailed.

The members receive an information booklet from the book club every month containing presentations and reviews of books, special offers, a complete stock list of the book club, etc. To order or cancel items, the members then fill out a form and return it to the book club. Some book clubs have set up a VC-service as an alternative to this procedure. The present system emulates such a service, but accepts voice input as well as DTMF keypad input.

4.2 Demonstrator Functionality

The basic task of the service is to allow members to dial up, and either place or cancel orders for books, records, posters and other items available to the members. It is presupposed that the member has the information booklet at hand, as all references to items from the user are by a four-digit number. The id-numbers are stated in the booklet. The system, however, uses both the id-number and the author and title when referencing items. This is done to eliminate misunderstandings, caused by e.g. an error by the user, or a system misrecognition. The advantage of this mode of communication is of course to reduce the vocabulary size for the speech recogniser, and hence improve recognition performance. The problem of updating (or adapting) the speech recogniser's vocabulary is also avoided. (As changes are made monthly this would have been a major task, unless solved automatically). The drawback is a more unnatural dialogue.

The demonstrator has a limited functionality at certain points. For the present, the speech synthesiser is not interfaced to the dialogue platform. Therefore, a set of predefined words and phrases are synthesised off line, and then concatenated into appropriate sentences at runtime. The cost of this is a less flexible system, but as the demonstrator is used with a fixed set of available items, it is not a major problem.

4.3 The Dialogue Design Process

As mentioned in the introduction, the dialogue design is based on an existing VR system. This is an unusual approach. The design process normally involves carefully planned series of interviews and simulation-(Wizard-of-Oz)-experiments [6]. The aim of this work is, however, to

investigate to which extent this process can be circumvented for simpler systems, and hence speed up the design phase.

Two sources of information have formed the basis of the dialogue design:

- The Book Club information booklet, containing a list of available items, procedures for ordering, etc. The booklet is important, as the dialogue presupposes that the user has it present when calling the service.
- A number of sample "dialogues" with the existing VR-service. A dummy account was set up by the publishing company, enabling the dialogue designer to try out the system in various scenarios. Five different scenarios were constructed and each was tried out a number of times. A record of each scenario was made, with e.g. the transaction time, whether the task was completed or not, etc.

Primarily based on the sample dialogues, specifications of the functionality of the system were made. The dialogue was designed from these specifications, in the form of a number of dialogue graphs, roughly corresponding to the tasks identified within the functionality specification.

A number of tasks were identified, and corresponding dialogue graphs were constructed. The most important of these are:

Main Choice. Starts by activating **Check Member** and then enters a loop, where the user is prompted for whether he/she wishes to order or cancel items, get an overview of current orders, or to end the session. Control is then transferred to the appropriate subtask.

Check Member. Asks for the member's number, and checks with the domain database. **Check Member** is performed at the beginning of the dialogue and is a subtask to **Main Choice**

Help provides instructions to the user, when activated. Depending on the current dialogue context, specific instructions are given for the actual task. The user is then prompted for more help and given general information on the system use. When activated from **Main Choice**, general information is given. Help is a subtask to all other tasks.

Order, **Cancel** and **List** are subtasks to **Main Choice.** Their tasks is to handle ordering, to cancel previous orders, and to give an overview of orders.

The dialogue initiative is always with the system. The effect of this is, that the dialogue becomes somewhat stereotype, with a series of question-answer sequences. The user is only allowed to supply the information, he/she has been directly prompted for. Any other information is discarded by the system. The only point where the user is allowed the initiative is when requesting help. To compensate for the restrictions imposed by this, the user is allowed some freedom in the formulation of his/hers utterances to the system. This is accomplished by utilising word-spotting techniques in the speech recogniser.

4.4 Dialogue Implementation

The dialogue description, now represented as a set of dialogue graphs, is implemented in the Dialogue Description Language (DDL) described in Section 3 and [1], [2], [3]. The basic principle of the formalism is, that the dialogue graphs are transferred into a graphical representation, roughly analogous to the graphs. As the level of abstraction is lower, the DDL-representation will contain more detailed information, and comprise more nodes than the dialogue graphs. Figures 2 and 3 shows the dialogue graph for the task **Main Choice** and the corresponding DDL-implementation.

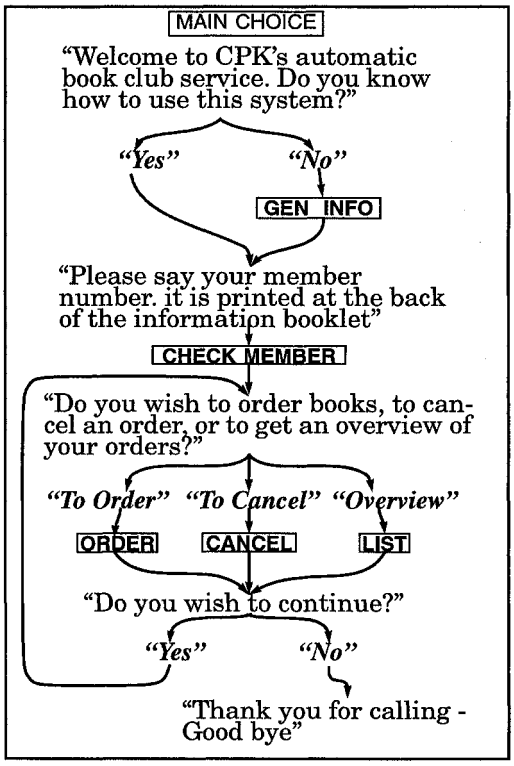

*Figure 2. Dialogue Graph for the subtask **"Main Choice"***

In the dialogue graph in Figure 2, subtasks are depicted as boxes with boldface captions. Text in *italics* is spoken input from the user, and ordinary

typeface is demonstrator output. After each output utterance, the system enters a *wait-state*, and waits for input from the user. Depending on the input, an arc is chosen and the dialogue progresses to the next system utterance, action or subtask. Wait-states are not shown explicitly on Figure 2.

This graph implements the functionality described in section 4.2. The core is the loop shown in the lower half of the figure, where control is transferred to the appropriate subtask according to the answer from the user. Figure 3 shows the DDL-implementation of the dialogue graph. The main loop can again be identified in the lower half of the figure:

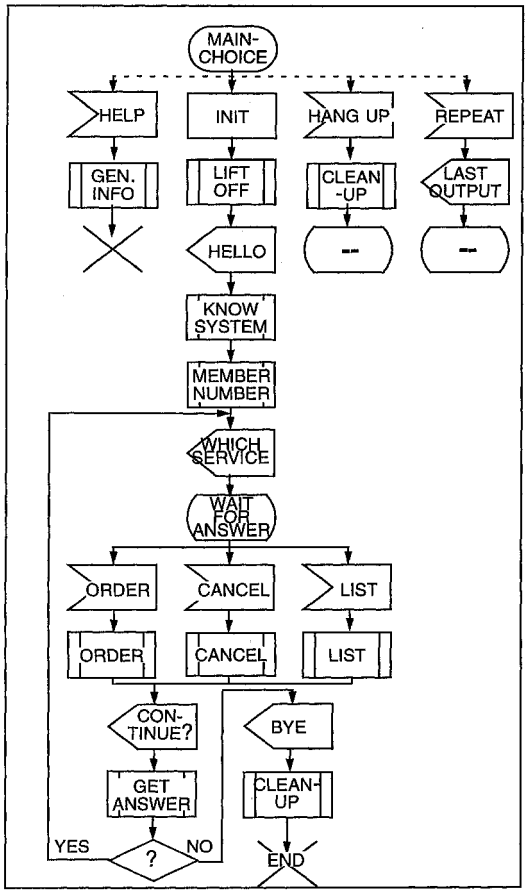

Figure 3.DDL-implementation of the Dialogue graph shown in Figure 2.

On the diagram, boxes with arrows pointing inwards denote input to the system, whereas an arrow pointing outward denotes an output from the system. A rectangular box with vertical lines denotes a procedure call (subtask). The symbols form an extended subset of the SDL symbols [8]. As can be seen by comparison, the graphical representation of the dialogue structure corresponds closely to the dialogue graphs. The branches on the upper part of the diagram are default-handlers, handling error situation, or special tasks, such as giving help instructions.

5. Discussion

The paper has presented the notion of utilising a generic dialogue platform as an application generator for spoken dialogue systems. A simple task has been chosen and implemented, using the DDL-formalism. As demonstrated in section 4, DDL-diagrams correspond closely to the dialogue graphs. The fact that the GDP makes devices, interfaces etc. transparent to the dialogue developer is very important. It ensures that prototypes can be constructed and evaluated rapidly.

The dialogue description for the present demonstrator comprises 13 dialogue graphs, which in turn are implemented as 20 DDL-diagrams with a total of 300 SDL-symbols. About 1.5 man-month was used to develop the prototype (excluding generation of speaker independent speech models).

6. References

[1] A. Baekgaard: "Dialogue Description Language", Doc. No. STC-DDL-2.1 ESPRIT SUNSTAR Project. CPK, Aalborg University 1992.

[2] A. Baekgaard, P. Dalsgaard: "Tools for designing dialogues in speech understanding environments". ICLSP-90, Kobe, Japan 1990.

[3] A. Baekgaard, "The Generic Dialogue System Platform", Report 10*, (forthcoming). CPK, Aalborg University, Denmark 1994.

[4] B. Lindberg et al.: "An integrated dialogue design and continuous speech recognition system environment". ICLSP-92, Alberta, Canada 1992.

[5] P.B. Nielsen, A. Baekgaard: "Experience with a dialogue description formalism for realistic applications". ICLSP-92, Alberta, Canada 1992.

[6] L. Dybkjær, H. Dybkjær: "Wizard of Oz. Experiments in the Development of the Dialogue Model for P1", Report 3a*, CCS, University of Roskilde, Denmark, February 1993.

[7] H. Dybkjær, L. Dybkjær: "Representation and Implementation of Spoken Dialogues", Report 6b*, May 1994, CCS, University of Roskilde, Denmark, February 1993.

[8] Belina, F., Hogrefe, D.: "The CCITT Specification and Description Language SDL". in "Computer Networks and ISDN Systems no. 16 (1988/89)", North-Holland pp. 311-341.

[9] B. Lindberg, J. Kristiansen, "Real-Time Speech Recognition within Dialogue Systems", Report 8*, (forthcoming). CPK, Aalborg University, Denmark 1994.

[10] B. Music, L. Offersgaard: "The NLP module", Report 7*, (forthcoming). CST, Center For Language Technology, Copenhagen, Denmark. May 1994.

*) Report series from the Danish research project "Spoken Language Dialogue Systems".