ISCA Archive
http://www.isca-speech.org/archive

First International Conference on
Spoken Language Processing (ICSLP 90)
Kobe, Japan
November 18-22, 1990

TOOLS FOR DESIGNING DIALOGUES IN SPEECH UNDERSTANDING INTERFACES.

Anders Baekgaard & Paul Dalsgaard
Speech Technology Centre
Aalborg University, Denmark

ABSTRACT

The ESPRIT project 'Integration of Speech Understanding Interfaces ' (SUNSTAR) concentrates on the design of a Dialogue System and the integration of existing off-the-shelf speech input and output devices into a number of practical applications. The communication between the speech I/O devices and a specific application is managed by a Dialogue System, which consists of a number of modules the aim of which are to guide speech activated I/O messages between the user and the application in a user-friendly and meaningful way, to forward user requests to the application, and to send back responses from the application to the user. Furthermore, the Dialogue System is supervising the state of functionality of the entire man-machine system.

This paper focuses its attention on the development of the individual modules of the Dialogue System, which consists of the Dialogue Description Language including a common Dialogue Design Tool and an Interpretation and Control Module. The Dialogue Design Tool is an essential part of the Dialogue System. It is used by the Dialogue Developer for assistance during the specification of a specific application dialogue.

The SUNSTAR project has partners from industry, telecommunication organisations and universities from five European countries, and the project is focused on establishing a number of practical speech I/O activated applications for the professional and public domains.

1. INTRODUCTION

Speech technology modules have reached a stage of practical applicability where it becomes realistic and profitable to initiate efforts aiming at the establishment of large scale applications. These are to be identified on the basis of an analysis of technical approach, market risks and attractiveness.

In the ESPRIT-II programme, a large scale project on 'Integration of Speech Understanding Interfaces' - the SUNSTAR project - has been initiated for that purpose. The aim of the project is to establish a number of practical and profitable market oriented applications, where a user will have the opportunity to be connected to a specific application via a user cooperative and assisting Dialogue System.

Work in the initial phase of the project is concentrated on the use of existing off-the-shelf speech I/O devices and on the establishment of a prototype Dialogue System in order to verify the feasibility of combining these elements into a number of selected applications. These are identified from the professional domain (Office and Workstation systems) and from the public domain (Public Telephone Services and Private Automatic Branch Exchanges). During the initial phase the applications will be established in a number of demonstrator prototypes by which it is possible to test realistic user environments. This means that especially the applied speech input devices must be able to recognise speech uttered under practical environmental conditions, i.e. the speech signal may be contaminated by noise from the surroundings and/or superimposed echo signals from a simultaneously functioning speech output device or the telephone line itself.

The Dialogue System will be designed in such a way that the user is assisted in his/her efforts to communicate with the application and by taking into consideration that adverse conditioned user-input often exists in practice. A further optimisation of the functionality of the Dialogue System will be obtained by taking also the experience of the user into consideration. A less experienced user must be guided more carefully through the communication with the application than a more experienced. This aspect, however, of the Dialogue System design of the SUNSTAR project is not dealt with in this paper, it is only mentioned here in order to point at one example out of a number of external conditions, which has to be taken into consideration during the design of the Dialogue System.

The Dialogue System consists of two modules, the Dialogue Description Language (DDL) containing the Dialogue Description and the Dialogue Design Tool (DDL-Tool), and the Interpretation and Control Module (ICM).

The integration of a Dialogue System with a specific application involves two steps. Firstly, the Dialogue Developer must give a complete specification of the dialogue structures for that application. The Dialogue Structures (identified in Figure 1 as DDL/SDL, DDL/FL and DDL/TL) are explained in details in section 4. The work of the Dialogue Developer in specifying the dialogue structures is called 'Dialogue Design', which is performed in

a separate session, and the outcome of the Dialogue Design is called a 'Dialogue Description'. Secondly, the Dialogue Description is loaded into the Dialogue System, where it is processed by the ICM, which controls the interaction between the user and the application.
The structure of the flow from Dialogue Design via parsing to interpretable code controlling the dialogue is shown Figure 1.

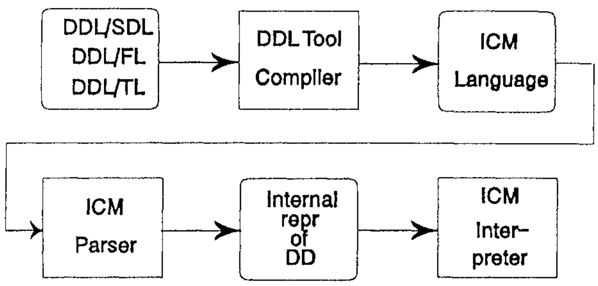

Figure 1. Structure of flow in the Dialogue System

In the following sections the Dialogue Description Language and the Dialogue Design Tool is described in some details. First, however, the general framework for the SUNSTAR demonstrators is presented.

2. DEMONSTRATOR ARCHITECTURE

Figure 2 shows the general architecture of a SUNSTAR demonstrator. This includes the I/O modules, the application and the Dialogue System. To the left are shown examples of a number of possible I/O modules. At the upper right of the figure is shown the application, which may belong either to the professional or the public domain. Also at the right the Interpretation and Control Module (ICM) is shown. The individual parts of the demonstrator system are interconnected via common interfaces, and all connected to the overall SUNSTAR Communication Manager (SCM), which manages system initialisation and controls the communication within the system.

Almost all commands and messages in the demonstrator are interpreted by the ICM, which therefore is of central importance for the functionality of the demonstrator and user interaction. The user is able to communicate with the system via different types of input and output devices. The actual devices at the disposal of the user depend on the application in question. E.g. in a public telephone application, DTMF and speech input are the only input devices available and speech output the only output device available.

3. AIMS OF DIALOGUE DESIGN

The Dialogue Developer must specify the Dialogue Description in a separate session prior to the set-up of any new application. The aim of the Dialogue Design is two fold. Firstly, a properly designed dialogue must aim at the most flexible and user-friendly interaction with the application. Secondly, the dialogue structures (see section 4) may be utilised to increase the performance of the chosen

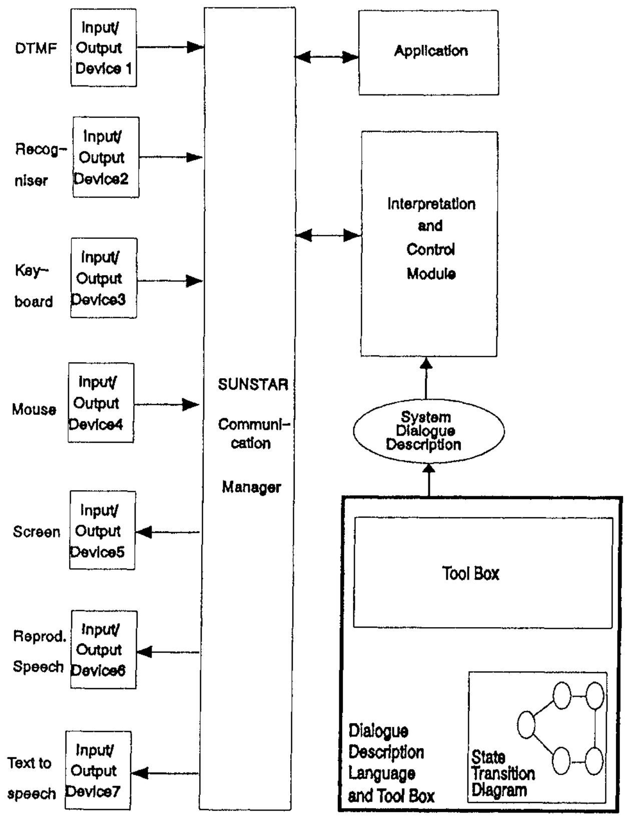

Figure 2. Architecture of SUNSTAR Demonstrator

input devices, and must be able to handle e.g. misrecognitions coming from the speech input device.

A Dialogue Design must take into consideration the specific characteristics of a user, the so-called user profile. A user profile may be a static description acquired from an application, or it may be dynamically estimated during the users interaction with the application. Many of the parameters in the user profile may be obtained by the ICM, counting the number of errors made by the user, counting the number of help requests, measuring latency time, etc. These parameters are then mapped into the user profile.
The dialogue must be designed to accommodate the user profiles. For example, an experienced user may be handled such that informative messages are short or totally omitted, whereas an unexperienced user may be faced with more detailed informative messages.

The dialogue structures must be flexible in such a way that a user may accomplish the same function in several ways. E.g. a user may want to arrange for a telephone calling at the hour 7:15. This can be performed by uttering single words in several different ways, e.g. seven-fifteen, a-quarter-past-seven or fifteen-minutes-past-seven all having the same meaning. In other words, the user-input may have several different syntactical representations of the same semantic representation. Hence, the Dialogue Description only needs to contain one specification of how to handle a given semantic representation supplied with a grammar definition, which specifies the various syntactical representations for that semantic representation.

For the initial phase of the SUNSTAR project, only speaker dependent and speaker independent isolated word recognisers are used. Connected word recognisers will be adopted later in the project.

4. TOOL AND LANGUAGE FOR DIALOGUE DESCRIPTION

In the SUNSTAR project, various methods for representing a dialogue description have been investigated [1, 2]. Based on that it has been decided to adopt the principles known from the state transition diagram formalism. This formalism is based on a graphical representation technique, which is well suited for supplying the user with a detailed survey of the flow of the dialogue. In addition, the state diagram formalism is easy to use for the Dialogue Developer compared to e.g. a BNF notation or a conventional programming language. Therefore, a graphical oriented DDL-Tool is designed in the SUNSTAR project in order to provide the Dialogue Developer with drawing facilities to be used during the Dialogue Design.

The DDL provides facilities for describing important aspects of a user dialogue. These are:

- high level control of devices
- communication with applications
- simple computations
- simple interpretation of user-input
- generation of user output
- management of dialogue history
- management of user profile
- management of semantic structures
- error handling

Additionally, facilities are provided for debugging, tracing and logging.

The dialogue description language (DDL) consists of three language levels: the graphical (DDL/SDL), the frame (DDL/FL) and the textual level (DDL/TL), cf. figure 1.

The graphical level represents the flow of the dialogue, i.e. the control of the order in which all events and actions in the dialogue occur. A typical flow sequence may be : 1) 'Prompt the user for input', 2) 'wait for user input' and dependent on input 3a) 'enter a subdialogue' or 3b) 'give immediate response', as indicated in Figure 3.

The frame level DDL/FL and the textual level DDL/TL represent the lower levels of the dialogue, which includes all the actual values of a number of different parameters and values. E.g. a user-input may originate from recogniser no. 4, being the single word 'help', having the recognition score 0.9.

The layout of the graphical level is based on the 'Specification and Description Language (SDL)', which is standardised by CCITT [3]. The SDL consists of a number

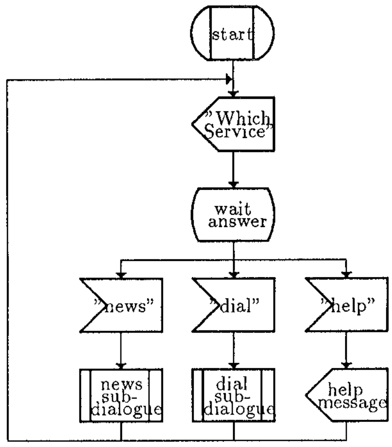

Figure 3. Example of Dialogue Description Flowgraph

of graphical symbols representing all events and actions, which may occur in a dialogue (e.g. user-input, user-output, application-input, decision, waiting-state, etc.). The symbols are connected by arcs which indicate the flow of the dialogue. For the purpose of Dialogue Design in the SUNSTAR project only a subset of the standardised SDL is used, and a few additional symbols have been defined.

To each of the SDL symbols is attached a Frame Level part DDL/FL, which specifies the lower levels of the dialogue. The DDL/FL consists of a number of predefined and Dialogue Developer defined frame structures. A frame structure consists of a number of key-value pairs. In cases where the Dialogue Developer has to specify the use of e.g. 'text-to-speech' output (tts), the frame structure might be specified as:

tts::

dev	*: english_tts*	(specifies output device)
txt	*: "Hello world"*	(specifies the text)
voc	*: female*	(specifies the voice type)

If any of the key-value pairs are not specified by the Dialogue Developer, default values will be used. Different types of frame structures are defined for different kinds of SDL symbols, i.e. the 'text-to-speech' frame can only be used in connection with a User-output symbol. The frame structure attached to e.g. a Decision symbol will define which value and conditions are to be used for the decision.

The DDL/FL is automatically transformed into the Textual Level DDL/TL by the DDL-Tool. Thus, like the DDL/FL, a DDL/TL part is associated with each of the SDL symbols. The DDL/TL provides facilities to specify computations, to specify various actions, etc. It is possible to define constants, types, variables, objects and operators in the DDL/TL. Objects are similar to frame structures, but may have operators bound to them.

All frame structures in the DDL/FL have a corresponding textual representation, which is accessible in the DDL/TL. The corresponding DDL/TL representation of the 'text-to-speech' frame shown above might be:

obj tts_parent(*dev english_tts, txt "Hello world", voc female*)

If necessary, it is possible for the Dialogue Designer to access the DDL/TL directly in order to make special editing, which is not possible in the DDL/FL.

Devices, commands and events are modelled as frame structures/objects. A user profile may be modelled as an object with a number of keys, which correspond to the actual parameters relevant for the specific dialogue.

As mentioned above, syntactically different representations of user-input may result in the same semantic representation. For this reason, only the semantic representation is used during the dialogue specification. The different syntactical representations are specified in a special syntax diagram. The syntax diagrams are specified in a way similar to the Dialogue Descriptions, using a subset of the SDL symbols. The syntax must be specified to such a level of detail, where only elementary events (single words) are involved. This corresponds to a graphical representation of a phrase structure except that iterations and recursions are not allowed. All elementary events are described in a lexicon stating the syntactic category, the semantic category and the semantic value. Using the syntax descriptions and the definitions in the lexicon, the ICM is able to parse the user-input accordingly, and to generate the semantic representation to be used in the dialogue description.

The DDL has a facility for handling certain events "by default" using special default handlers. These are setup globally or locally in procedures. If an event occurs that is not explicitly described in an input symbol, it is tested whether a default handler can handle the event.

5. FURTHER FACILITIES OF THE DDL-TOOL

The DDL-Tool has a number of facilities added aiming at the optimal flexibility in use.

The DDL-Tool allows the Dialogue Developer to establish a Dialogue Description as a number of small diagrams, which are linked together with '**procedure definition**' and '**procedure call**' symbols. Therefore, a well structured Dialogue Description is easy to survey and maintain. The DDL-Tool provides functions which supports this structuring approach. For example, a procedure definition can be displayed simply by clicking at a corresponding '**procedure call**' symbol and a menu entry.

The DDL-Tool has facilities for:

- printing diagrams
- performing syntax check
- annotate comments to DDL files, e.g. maintenance log

An extensive set of graphical manipulation functions helps the Dialogue Developer to produce professional quality layout of diagrams.

6. CONCLUSIONS AND FUTURE WORK

The present stage of development of the dialogue system has shown that the introduction of a graphical oriented tool for defining dialogue descriptions is very useful. It gives a fast overview of the dialogue, which is defined for a specific application, and it allows for the Dialogue Developer to make experiments with different examples of dialogue structures before a spefic application is being applied in practice.

The run-time dialogue system will execute on a SPARC 1 workstation on which machine also all Dialogue Design is performed. The host SPARC-station is connected via TCP/IP to one or several PC's in which e.g. the speech I/O boards are placed, and to the application, which might also be running on the SPARC-station or in separate hardware.

Immediate future work is planned in two areas. Firstly, a next generation recognisers to be used will be able to recognise connected words in a speaker-independent way. Therefore, the present capabilities of handling dialogues based on recognition of isolated single words will be extended such that also dialogues bases on connected speech will be possible. Secondly, grammar definitions are presently very restricted because syntactical descriptions are not allowed to involve iterations or recursions. This restriction will be eliminated to allow more natural user-input.

7. ACKNOWLEDGEMENT

The authors wish to thank the Project Management Committee for their request to publish the present status of this essential element of the SUNSTAR project. We would also like to thank our colleague Jens Duhn for his assistance in formulating the paper.

8. REFERENCES

1. Roger W Ehrich, Robert C Williges. Human-Computer Dialogue Design, Elsevier 1986.

2. H Rex Hertson, D Hix. Human-Computer Interface Development: Concepts and Systems. ACM Computing Surveys, Vol 21, Number 1.

3. CCITT Recommandation Z.100: Specification and Description Language SDL, APIX-35, 1988.