Lyrebird™: Developing Spoken Dialog Systems using Examples

Bradford Starkie, Greg Findlow, Khanh Ho, Alvaro Hui, Lawrence Law, Liron Lightwood, Simon Michnowicz, and Christian Walder.

Telstra New Wave Pty Ltd, Box 249 Rosebank MDC 770 Blackburn Road Clayton Victoria, Australia 3168
{Brad.Starkie, Greg.Findlow, Khanh.Ho, Alvaro.Hui, Lawrence.Law, Liron.Lightwood,Simon.Michnowicz, Christian.Walder}@team.telstra.com

Abstract

An early release software product for the rapid development of spoken dialog systems (SDS's), known as Lyrebird™ [1][2][3], will be demonstrated that makes use of grammatical inference to build natural language, mixed initiative, speech recognition applications.

The demonstration will consist of the presenter developing a spoken dialog system using Lyrebird™, and will include a demonstration of some features that are still in the prototype phase.

Using Lyrebird™, developers build a spoken dialog system with the following steps:

1. Firstly the dialog is specified. This is typically developed using a drag and drop GUI that defines a computer directed menu driven application (See Figure. 1). Alternatively a wizard can be used whereby the developer defines the application in terms of slots that need to be filled.
2. The dialog description is then generalised to include mixed initiative input. Starting grammars are automatically created that cover computer directed input.
3. The developer is then prompted for text describing selected phrases that a speaker would use when interacting with the developed SDS (See Figure 2.) Mixed-initiative natural-language grammars suitable for use with a speech-recognition system are then inferred, using grammatical inference.

If desired, the developer can subsequently improve the system under development, using a range of interfaces. For instance:

1. The developer can teach Lyrebird how the speaker will behave using a tool known as the simulator. The simulator enables the user to interact with the application under development using text. (See Figure 3.) Out of grammar phrases are tagged using either robust parsing, or manual input from the developer.
2. The developer can teach Lyrebird how the system should behave using a tool known as the scenario editor (See Figure 4.) Using the scenario editor sample interactions are presented to the developer in a text format similar to that used

to describe plays. Users can modify prompts, speech recognition grammars and dialog simply by modifying the scenarios.

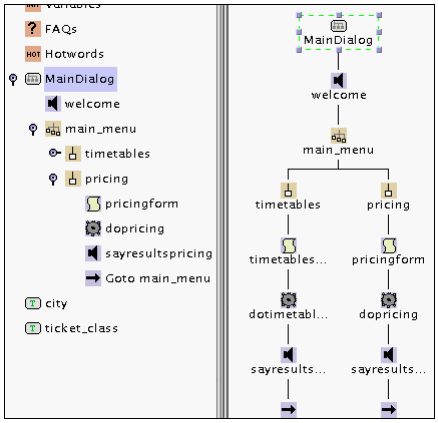

Figure 1. A menu driven dialog description created using Lyrebird™.

```
> Prompt> please say one of timetables or pricing (note please respond with a phrase for
jumping to the timetablesform form)
> Instructions> Please enter a phrase which contains the following slot values:
>                 Departure_city : salt lake city
>                 Destination_city : helsinki
>                 Date : a week from sunday
>                 Time : five a m
> Your Input>
> what flights fly from salt lake city to helsinki at five a m on a week from sunday
>
> Grammar Identifier> .Form_timetablesform
> Prompt> please say one of timetables or pricing (note please respond with a phrase for
jumping to the timetablesform form)
> Instructions> Please enter a phrase which contains the following slot values:
>                 Departure_city : salt lake city
>                 Date : a week from sunday
> Your Input>
> what flights fly from salt lake city on a week from sunday
> commit
>
> Grammar Identifier> .Form_timetablesform
> Prompt> please say one of timetables or pricing (note please respond with a phrase for
jumping to the timetablesform form)
> Hypothetical Response> what flights fly from milwaukee to amsterdam at two fifty nine
> Question> Is this a suitable response?
```

Figure 2. Example phrases being entered into Lyrebird™ by a developer.

```
tags, or select the manual entry option to input the correct tag values yourself.
> PROMPT: welcome to the airline application.
> PROMPT: please say one of timetables or pricing
>
> REPLY  >
> how much does a ticket from melbourne to sydney cost
>
>        Please enter the number corresponding to your selected option:
>        0. Re-enter phrase
>        1. Manually enter in attributes
>        2. airline#.Form_pricingform { destination_city="sydney" depature_city="melbourne"}
(p=0.842105)
>        3. airline#.Form_timetablesform { destination_city="sydney"
depature_city="melbourne"} (p=0.157895)
>        4. airline#.Question_operator  (p=3.30838e-48)
>        5. airline#.Quit  (p=2.31587e-48)
>        6. airline#.Question_hello  (p=1.65419e-48)
>        7. airline#.Menu_main_menu_pricing  (p=3.30838e-49)
>        8. airline#.Menu_main_menu_timetables  (p=3.30838e-49)
> YOUR CHOICE >
> 2
> PROMPT: please say the ticket class
>
> REPLY  >
> economy
> PROMPT: a economy flight from melbourne to sydney will cost five hundred dollars
> PROMPT: please say one of timetables or pricing
>
> REPLY  >
```

Figure 3. Example phrases entered into Lyrebird by a developer as seen in the "Simulate and Refine" window of Lyrebird™.

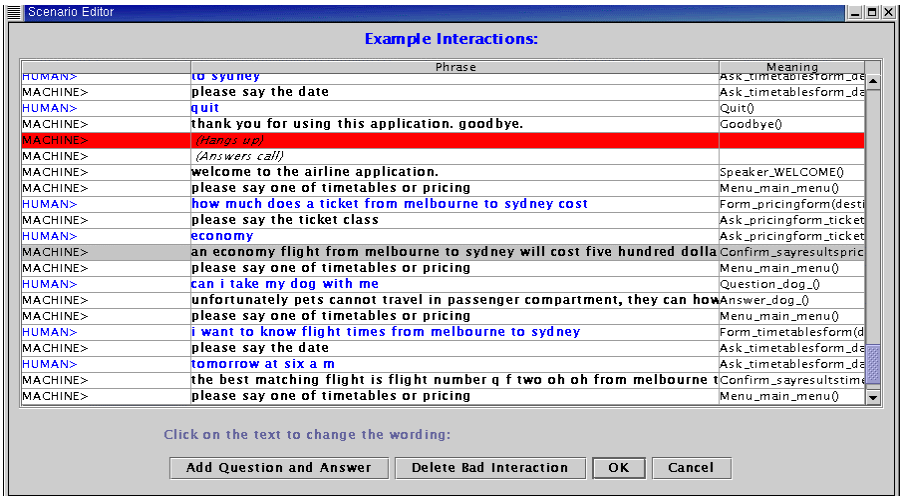

Figure 4. Example behaviours being modified by the developer in Lyrebird™ as seen in the "Scenario Editor" window.

A significant benefit of using Lyrebird™ is that developers don't need to know anything about speech recognition grammars or the underlying VXML used to implement the spoken dialog system. Instead the developers are presented with and present to Lyrebird™, examples of how the application should behave. In this respect Lyrebird can be seen to be a "Programming by Example" system [4]. In addition the use of Lyrebird™ can result in significant reductions in the time required to develop an application [1].

References

1. Starkie, Bradford C., 2002, Inferring Attribute Grammars with Structured Data for Natural Language Processing, in: *Grammatical Inference and Applications. Sixth International Colloquium*, ICGI-2002, pp 1-4 Berlin: Springer Verlag.
2. Starkie, Bradford C., 2001. Programming Spoken Dialogs Using Grammatical Inference, in *AI 2001: Advances in Artificial Intelligence, 14th Australian Joint Conference on Artificial Intelligence*, pp 449-460, Berlin: Springer Verlag
3. Starkie, Bradford C., 1999. A method of developing an interactive system, International Patent WO 00/78022.
4. Cypher, Allen, 1993. *Watch what I do: programming by demonstration*, Cambridge Massachusetts: MIT press.

TM – Trade mark applied for by Telstra New Wave Pty Ltd.