

CLASSiC

D3.6: User simulations of different types of Appointment Scheduling and Self-Help user

Stéphane Rossignol, Srinivasan Janarthanam,
Xingkun Liu, Olivier Pietquin, Michel Iannotto

Distribution: Public

CLASSiC

Computational Learning in Adaptive Systems for Spoken Conversation
216594 Deliverable 3.6

December 2010



Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development



The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	216594
Project acronym	CLASSiC
Project full title	Computational Learning in Adaptive Systems for Spoken Conversation
Instrument	STREP
Thematic Priority	Cognitive Systems, Interaction, and Robotics
Start date / duration	01 March 2008 / 36 Months

Security	Public
Contractual date of delivery	M30 = August 2010
Actual date of delivery	December 2010
Deliverable number	3.6
Deliverable title	D3.6: User simulations of different types of Appointment Scheduling and Self-Help user
Type	Prototype
Status & version	Draft 1.0
Number of pages	12 (excluding front matter)
Contributing WP	3
WP/Task responsible	SUPELEC
Other contributors	
Author(s)	Stéphane Rossignol, Sridhar Janarthanam, Xingkun Liu, Olivier Pietquin, Michel Iannotto
EC Project Officer	Philippe Gelin
Keywords	Spoken dialogue systems, dialogue management

The partners in CLASSiC are:

Heriot-Watt University	HWU
University of Cambridge	UCAM
University of Geneva	GENE
Ecole Supérieure d'Electricité	SUPELEC
France Telecom/ Orange Labs	FT
University of Edinburgh HCRC	EDIN

For copies of reports, updates on project activities and other CLASSiC-related information, contact:

The CLASSiC Project Co-ordinator:
Dr. Oliver Lemon
School of Mathematical and Computer Sciences (MACS)
Heriot-Watt University
Edinburgh
EH14 4AS
United Kingdom
O.Lemon@hw.ac.uk
Phone +44 (131) 451 3782 - Fax +44 (0)131 451 3327

Copies of reports and other material can also be accessed via the project's administration homepage,
<http://www.classic-project.org>

No part of this document may be reproduced or transmitted in any form, or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission from the copyright owner.

Contents

- Executive Summary 1
- 1 Introduction 2
- 2 Simulations for experts/novices and users “in-between” on a knowledge spectrum 3
 - 2.1 Knowledge profiles 3
 - 2.2 Dialogue example with an intermediate user 4
- 3 Appointment scheduling task – Bayesian-Network based user simulator 6
 - 3.1 Description of the system 6
 - 3.2 Dialogue example 8
- 4 Description of the software associated with the prototypes 10
 - 4.1 Self-Help User Simulations 10
 - 4.2 Appointment Scheduling User Simulations 10
- Bibliography 12

Executive summary

This document is a short report to accompany the prototype deliverable 3.6, due at month 30 of the CLASSiC project. The prototype focuses on user simulations which can represent different types of user, in both the Appointment Scheduling and Self-Help domains.

This deliverable has been delayed to month 33 due to the time required to analyse data from WP 5 which was delivered later than planned. Publications related to this deliverable are presented in the Appendix. They are available at www.classic-project.org

1 Introduction

This document describes the prototype deliverable D3.6, due at month 30. The contribution consists of two parts: 1) Bayesian simulations for experts/novices and users “in-between” on a knowledge spectrum for the Self-Help task; and 2) the Dynamic Bayesian Network (DBN) simulator approach, applicable in the Appointment Scheduling domain (the simulator from Supélec being integrated within the dialogue system from the University of Cambridge). The prototypes are available on Supelec’s Forge(http://foundry.supelec.fr/svn/classic/trunk/cued_dialog_system_task_AS/).

2 Simulations for experts/novices and users “in-between” on a knowledge spectrum

Users in the technical Self-Help task can be characterised as having different levels of domain knowledge. A dialogue system should be able to identify the domain expertise of a user and adapt to him/her during the course of interaction. The Self-Help dialogue system was trained to learn and identify the domain knowledge level of the user and adapt to him/her by choosing appropriate referring expressions to refer to domain objects. The system has to choose between jargon expressions and descriptive expressions. While expert users appreciate jargon expressions, novice users find descriptive expressions more helpful to identify the domain objects referred to in the system’s instructions. The dialogue state of the system consists of a user model that represents current user’s domain knowledge. The user model is a vector of variables – one for each object in the domain. Each of these variables represents the system’s belief regarding the user’s knowledge of jargon expressions for a particular domain object. All of the variables are initially set to a value “unknown” signifying the fact that at the beginning of the conversation the system is unaware of the user’s domain knowledge. The natural language generation (NLG) module of the dialogue system has a referring expression generation (REG) policy that maps each state of the user model to the choice of referring expression for each domain object. Our objective is to learn such a REG policy using simulation based reinforcement learning (RL) techniques. In order to learn an adaptive REG policy, we developed a two-tier user simulation, that simulates the dialogue behaviour of different types of users (i.e. novice, expert, intermediate, etc) in a technical support task.

We presented a version of the two-tier user simulation model in deliverable D3.3 (“Simulated users for training NLG”). In this deliverable, we now focus on how the user simulation model is able to simulate different types of users.

2.1 Knowledge profiles

Complementary to the user action selection models presented in the previous section, we also model the domain knowledge of the users in the simulation module. This fulfils our requirement of simulating users with different domain knowledge levels. During training, the dialogue system interacts with the user simulation many times producing many dialogues. At the start of each dialogue, we set the initial knowledge of the user simulation to one of the several knowledge profiles. Once instantiated, the simulation produces a dialogue behaviour that is consistent with its knowledge profile. For example, it behaves like a novice user asking a lot of clarification requests for jargon expressions when instantiated with a novice profile. The initial knowledge base ($DK_{u,initial}$) for 5 different users is shown in table 1. A novice user knows only “wall phone socket”, and an expert knows all the jargon expressions. Between these two extremes, there are three intermediate profiles as well.

The knowledge of a jargon expression x is represented by the corresponding domain entity (R_x) in the profile. We use Y in the table to denote the user’s knowledge of the jargon expression. The model assumes that users can interpret the descriptive expressions and resolve their references. Therefore, they are not explicitly represented. We use these five stereotype knowledge profiles as initial domain knowledge levels of users simulated during training and evaluation phases. By using different profiles, the models produce a consistent behaviour of different types of users at various domain expertise levels.

These knowledge profiles were learned from the pre-task knowledge test of our corpus that we collected from 17 real users using a Wizard-of-Oz setup (see deliverable D6.1.1). They data was clustered using

	Novice	Int1	Int2	Int3	Expert
Phone socket	Y	Y	Y	Y	Y
Livebox		Y	Y	Y	Y
Livebox Power socket		Y	Y	Y	Y
Livebox Power light		Y	Y	Y	Y
Power adaptor			Y	Y	Y
Broadband cable				Y	Y
Ethernet cable			Y	Y	Y
Livebox Broadband light					Y
Livebox Ethernet light				Y	Y
Livebox ADSL socket					Y
Livebox Ethernet socket				Y	Y
PC Ethernet socket			Y	Y	Y
Broadband filter					Y

Table 1: Domain knowledge of 5 different users

the *k-means* algorithm into 5 clusters to produce stereotypical knowledge profiles of users with different domain expertise levels. We used the centroid (i.e. center of the cluster) profiles as our stereotypes.

2.2 Dialogue example with an intermediate user

Sys: Are you ready?

Usr: Yes.

Sys: Do you have a **livebox** in the package?

Usr: Yes.

Sys: Do you have a **broadband cable** in the package?

Usr: Which one is the broadband cable?

Sys: The broadband cable is the thin black cable with colourless plastic ends.

Usr: Yes.

Sys: Do you have an **ethernet cable** in the package?

Usr: Yes.

Sys: Do you have a **power adaptor** in the package?

Usr: Yes.

Sys: Do you have a **broadband filter** in the package?

Usr: Which one is the broadband filter?

Sys: The broadband filter is the small white box that has two sockets and an phone plug.

Usr: Yes.

Sys: Good. We seem to have everything we need. Now follow my instructions to set up the connections. Tell me when you are ready.

Usr: I am ready.

Sys: Please disconnect the phone from the **phone socket** on the wall.

Usr: Done.

Sys: Please take the **power adaptor**.

Usr: Yes.

Sys: Please plug the **power adaptor** into the two-pin mains power socket.

Usr: Yes.

Sys: Please connect the cable of the **power adaptor** firmly into the **power socket** of the **livebox**. Observe the lights on the front panel of the **livebox**.

Usr: Yes.

Sys: Did all the lights on the front panel of the **livebox** turn on and go off after a few seconds.

Usr: Yes.

Sys: Place the **livebox** with the Orange label facing up. What is the status of the **power light** on the **livebox**?

Usr: Its flashing red.

Sys: Please plug one end of the **thin black cable with colourless plastic ends** into the **ADSL socket** on the **livebox**.

Usr: Yes.

Sys: Take the **small white box that has two sockets and a phone plug**.

Usr: Yes.

Sys: Please plug the other end of the **thin black cable with colourless plastic ends** into the **small white box that has two sockets and a phone plug**.

Usr: Yes.

Sys: Please plug the **small white box that has two sockets and a phone plug** into the **phone socket** on the wall.

Usr: Yes.

Sys: What is the status of the **second light from the left** on the **livebox**?

Usr: Its flashing red.

Sys: Please plug one end of the **ethernet cable** into the **square socket with red stripe** on the **livebox**.

Usr: Yes.

Sys: Please connect the other end of the **ethernet cable** into the **ethernet socket** on the back panel of your computer.

Usr: Yes.

Sys: What is the status of the **fourth light from the left** on the **livebox**?

Usr: Its red.

Sys: Please connect the phone cable into the **small white box that has two sockets and a phone plug** that you plugged into the phone socket on the wall.

Usr: Ok.

Sys: Good. We have now finished setting up your broadband Internet connection.

3 Appointment scheduling task – Bayesian-Network based user simulator

3.1 Description of the system

The Bayesian-Network based user simulator needed to be adapted to fulfil the requirements of the Appointment Scheduling task. The new model is presented on figure 1. The model has been integrated within the HIS, the dialogue management system from the University of Cambridge. It interacts with a Dialogue Manager designed and trained for the Appointment Scheduling task. The simulator and the dialogue manager are available on Supélec's Forge.

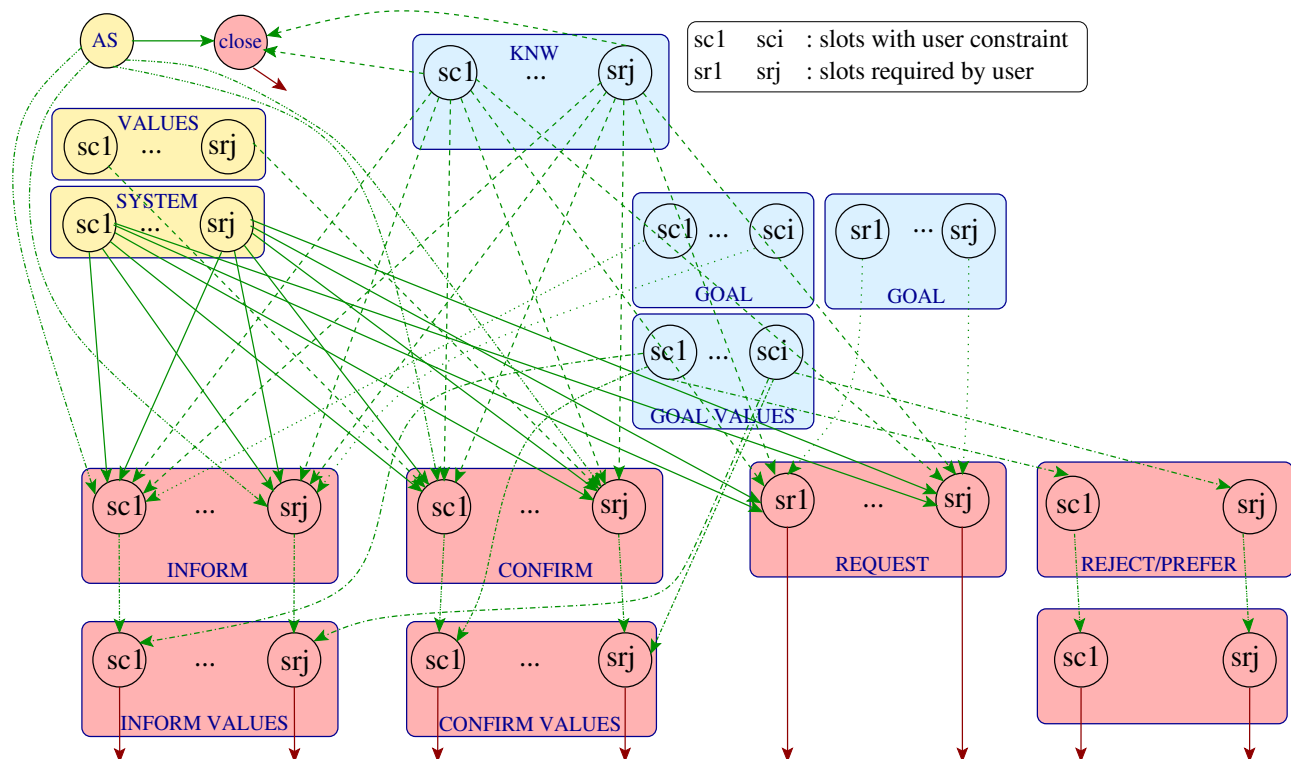


Figure 1: BN model for the appointment scheduling task; notice that the grounding component, which is present, is not represented here for readability reasons

The user simulator is able to adapt its goal in a more or less collaborative way:

- At the beginning of each dialogue, the dialogue manager provides a Calendar (that is to say a list of possible time slots), to the Simulator. As an example, see the table 2 (this example corresponds to the Calendar used when the dialogue in section 3.2 has been simulated).

Initially, the user goal is the first possible time slot. If the system refuses it, indicating that there is not technician available for this time slot, the user can switch to the second possible time slot (etc.). The user can be more or less collaborative: the probability of switching is fixed to a higher or lower value.

Today:	2010-Jul-12			
Day 2;	2010-Jul-13	Tuesday (am):	busy	[week 28]
Day 2;	2010-Jul-13	Tuesday (pm):	busy	[week 28]
Day 3;	2010-Jul-14	Wednesday (am):	busy	[week 28]
Day 3;	2010-Jul-14	Wednesday (pm):	busy	[week 28]
Day 4;	2010-Jul-15	Thursday (am):	busy	[week 28]
Day 4;	2010-Jul-15	Thursday (pm):	free	[week 28]
Day 5;	2010-Jul-16	Friday (am):	busy	[week 28]
Day 5;	2010-Jul-16	Friday (pm):	busy	[week 28]
Day 6;	2010-Jul-17	Saturday (am):	busy	[week 28]
Day 6;	2010-Jul-17	Saturday (pm):	free	[week 28]
Day 7;	2010-Jul-18	Sunday (am):	free	[week 28]
Day 7;	2010-Jul-18	Sunday (pm):	busy	[week 28]
Day 8;	2010-Jul-19	Monday (am):	free	[week 29]
Day 8;	2010-Jul-19	Monday (pm):	busy	[week 29]
Day 9;	2010-Jul-20	Tuesday (am):	busy	[week 29]
Day 9;	2010-Jul-20	Tuesday (pm):	busy	[week 29]
Day 10;	2010-Jul-21	Wednesday (am):	free	[week 29]
Day 10;	2010-Jul-21	Wednesday (pm):	free	[week 29]
Day 11;	2010-Jul-22	Thursday (am):	free	[week 29]
Day 11;	2010-Jul-22	Thursday (pm):	free	[week 29]
Day 12;	2010-Jul-23	Friday (am):	free	[week 29]
Day 12;	2010-Jul-23	Friday (pm):	free	[week 29]
Day 13;	2010-Jul-24	Saturday (am):	free	[week 29]
Day 13;	2010-Jul-24	Saturday (pm):	busy	[week 29]
Day 14;	2010-Jul-25	Sunday (am):	free	[week 29]
Day 14;	2010-Jul-25	Sunday (pm):	busy	[week 29]

Table 2: Calendar, indicating the available time slots

Internally, referring to the figure 1, the user goal comprises two goals: the Date (with values from 12 to 25) and the Time (with values equal to “am” or ”pm”). Both of them are “user constrained” goals.

- If the system refuses a user time slot, or if the user detects a grounding problem, the simulator sends a reject/ disprefer/ negate message to the system, in which it provides its list of impossible time slots.

This list can be more or less complete, reflecting a more or less collaborative user behaviour.

The user can be more or less collaborative in another way: a probability to send or not the reject/ disprefer/ negate message can be fixed.

- Increasing or decreasing the probabilities that the BN sends “inform” messages, allows the user to be more or less assertive.

3.2 Dialogue example

A dialogue examples is provided. The flag “Trained 2” indicates that the Conditional Probability Tables (CPTs) of the Bayesian-Network in use have been heuristically fixed by an expert and retrained using the Towninfo database (please see Deliverable D3.4 in order to get more insight into the techniques in use).

```
User goal: -- UNSTANDALONE -- Trained 2 --
  [slot_1] date:  fifteenth
  [slot_2] time:  pm
```

```
Initial knowledge: -- UNSTANDALONE --
  date:  low
  time:  low
```

The dialogue starts

----- New Turn -----

Message sent by the system to the simulated user:

```
system action = hello      num_slot = Date      slot_value = fourteenth
```

(current knowledge (before decision process) is:

```
[slot_1] date: 'low'      [slot_2] time: 'low')
```

Decision process

no grounding error detected for the current turn

Message from the simulated user (internal):

```
inform(slot_1='fifteenth')
```

Answers sent by the simulated user to the DM:

```
(0) inform slot_3 Thursday

(current knowledge (after decision process) is:
  [slot_1] date: 'low'    [slot_2] time: 'high')

----- New Turn -----
Message sent by the system to the simulated user:
  system action = request    num_slot = Time

(current knowledge (before decision process) is:
  [slot_1] date: 'low'    [slot_2] time: 'high')

Decision process
no grounding error detected for the current turn

Message from the simulated user (internal):
  inform(slot_2='pm')

Answers sent by the simulated user to the DM:
  (0) inform slot_2 pm

(current knowledge (after decision process) is:
  [slot_1] date: 'high'   [slot_2] time: 'high')

----- New Turn -----
Message sent by the system to the simulated user:
  system action = inform    num_slot = booking    slot_value = final

(current knowledge (before decision process) is:
  [slot_1] date: 'high'   [slot_2] time: 'high')

Decision process
no grounding error detected for the current turn

Message from the simulated user (internal):
  bye()

Answers sent by the simulated user to the DM:
  (0) bye

(current knowledge (after decision process) is:
  [slot_1] date: 'high'   [slot_2] time: 'high')
```

4 Description of the software associated with the prototypes

The prototypes consist of the following software:

4.1 Self-Help User Simulations

The Java code of two-tier Self-help user simulation model was described in sections 2.5 and 3.5 in deliverable D3.3 (Simulated Users for training NLG). The focus of this deliverable is on how the user simulation was manipulated to simulate the dialogue behaviour of different types of users (i.e. novice, expert, intermediate, etc). The five knowledge profiles (DK) are represented as vectors of 13 binary variables (1 - user knows jargon, 0 - user does not know jargon), one for each object in the domain in the same order as in table 1. At the start of each dialogue episode, one of the 5 profiles is randomly chosen as follows.

```
int[][] profile = {{1,0,0,0,0,0,0,0,0,0,0,0,0},
{1,1,1,1,0,0,0,0,0,0,0,0,0},
{1,1,1,1,1,0,1,0,0,0,0,1,0},
{1,1,1,1,1,1,1,0,1,0,1,1,0},
{1,1,1,1,1,1,1,1,1,1,1,1,1}};
```

```
current_profile = choose_random(profile);
```

4.2 Appointment Scheduling User Simulations

Introduction

This section describes code of the prototype deliverable consisting of the Bayesian Network based user simulator integrated within tHIS (Cambridge dialogue system). The task domain for this system is Appointment Scheduling, i.e., users try to negotiate a common date for the appointment. They can express their preferences and dispreferences as to their date. They can also change their mind or adjust their agenda constraints to the system agenda.

The HIS system consists of several modules, ranging from speech understanding and generation components to the central component, the dialogue manager (for a more detailed description of the HIS system, see [1]). In order to deal with the uncertainties due to speech recognition and understanding errors, the Dialogue Manager uses a Partially Observable Markov Decision Process (POMDP) approach. In this approach, a large amount of data is requested to train the Dialogue Manager. The trained data is obtained via a user simulator, which simulate the behaviour of an actual human users. In the following sections, a brief overview will be given of the different components in the current user simulator prototype. For a more detailed description of the Bayesian Network based user simulator, see [2], [3].

Interfacing the User Simulation from Supélec and the Dialogue System from Cambridge

The software for this prototype deliverable is organised in several directories.

- tHIS: C++ sources of dialogue management software, including:

- HISLib : the HIS dialogue manager
 - UMLib : user simulation and evaluation; the Bayesian-Based user simulator from Supélec is added in this directory
 - TDMan : the test harness
- atk: speech recognition and synthesis software
 - SemIO : C++ sources of SVM based semantic decoder software
 - etc.

The tHIS system to which the user simulator is interfaced is written in C++. The user simulator is written in C. This C code is embedded in a class constructor: *UMBNSSim::UMBNSSim(int trace)*.

User Simulator

The two open source librairies *smile* (definitions of the nodes and the arcs of a Bayesian Network) and *smilearn* (training of a Bayesian Network) are used. A grounding component is present and a EM based learning process of the BN parameters is used. The most important functions are:

- *void CreateNetworkHeuristics(void);void CreateNetworkTrained(void);*: create the Bayesian Network (nodes, arcs and probabilities).
- *void returngoal(char *listSlotGoal[2], char *listValueGoal[2], int numval[2]);*: returns the user goal, that is to say here the user agenda. The user agenda is determined by the tHIS system, which requires it in order to compute completion task measures; at the beginning of each dialogue, the tHIS system sends the current user goal to the user simulator. The arguments are the slots in the goal and value for each slot, this as lists as several date are possible.
- *int InferenceWithBayesNet(char sysmessage1[STRSIZE], char sysmessage2orig[NMAXSLTS][STRSIZE], char sysmessage3orig[NMAXSLTS][STRSIZE], answ **answers, double **probabilities, int *nslots);*: the inference with the Bayesian Network itself.
- *int networklearning();*: tool that allows the training of the probabilities.

Bibliography

- [1] S. Keizer F. Mairesse B. Thomson S. Young, M. Gašić and K. Yu. The hidden information state model: a practical framework for pomdp based spoken dialogue management. In *Computer Speech and Language*, 2009.
- [2] Stéphane Rossignol, Olivier Pietquin, and Michel Ianotto. Grounding simulation in spoken dialog systems with bayesian networks. In G. Geunbae Lee et al., editor, *Proceedings of the International Workshop on Spoken Dialogue Systems (IWSDS 2010)*, volume 6392 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 110–121, Gotemba (Japan), October 2010. Springer-Verlag, Heidelberg-Berlin.
- [3] Olivier Pietquin, Stéphane Rossignol, and Michel Ianotto. Training bayesian networks for realistic man-machine spoken dialogue simulation. In *Proceedings of the 1rst International Workshop on Spoken Dialogue Systems Technology (IWSDS 2009)*, Irsee (Germany), December 2009. 4 pages.