

CLASSiC

D5.4: Proof-of-concept CLASSIC Appointment Scheduling system (“System 2”)

Filip Jurčiček, Simon Keizer, François Mairesse, Kai Yu,
Steve Young, Srinivanan Janarthanam, Helen Hastie, Xingkun Liu,
and Oliver Lemon

Distribution: Public

CLASSiC

Computational Learning in Adaptive Systems for Spoken Conversation
216594 Deliverable 5.4

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	M33 = November 2010
Actual date of delivery	December 2010
Deliverable number	5.4
Deliverable title	D5.4: Proof-of-concept CLASSiC Appointment Scheduling system (“System 2”)
Type	Prototype
Status & version	Draft 1.0
Number of pages	19 (excluding front matter)
Contributing WP	5
WP/Task responsible	UCAM
Other contributors	HWU, UEDIN
Author(s)	Filip Jurčíček, Simon Keizer, François Mairesse, Kai Yu, Steve Young, Srinivanan Janarthanam, Helen Hastie, Xingkun Liu, and Oliver Lemon
EC Project Officer	Philippe Gelin
Keywords	Spoken dialogue systems, dialogue management, user simulation

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>

©2010, The Individual Authors.

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 ASR	2
3 Dialogue management	3
3.1 Slot names and slot values	3
3.2 Input dialogue act scheme	4
3.3 Output dialogue act scheme	5
3.4 Belief state update and the summary actions	6
3.5 Simulated user	7
3.6 Structure of an appointment scheduling dialogue	7
4 Spoken Language Understanding	7
5 Natural Language Generation	8
5.1 Handcrafted NLG baseline	9
5.2 Reinforcement learning based NLG	9
6 TTS	17
7 Software	17
8 Conclusions	18
Bibliography	19

Executive summary

This document is a report to accompany the Prototype deliverable D5.4, due at month 33 of the CLASSiC project. It describes the CLASSiC System 2 prototype, a French Appointment Scheduling dialogue system operating via telephone. Each module of the dialogue system is described in detail.

1 Introduction

This document describes the prototype deliverable consisting of the dialogue system developed by Cambridge University, Edinburgh University, and Heriot-Watt University for the Appointment Scheduling domain.

The systems were integrated on the ATK dialogue system platform provided by Cambridge University, which has also been used for CLASSiC System 1. For a specification of the architecture, see Deliverable D5.1.1. The dialogue system is a French system that clients of France Telecom can call to book appointments with a technician to fix their internet connection. To enable natural language interaction over the phone, a VoIP interface is used.

Similar to the CLASSiC System 1, the system consists of 5 distinct modules: automatic speech recognition (ASR), semantic decoding or spoken language understanding (SLU), dialogue management (DM), natural language generation (NLG) and speech synthesis or text-to-speech (TTS). These modules have been developed by various partners of the CLASSiC project:

Components	Contributors
ASR	Cambridge University
SLU	Cambridge University
DM	Cambridge University
NLG	Cambridge University and Heriot-Watt / Edinburgh University
TTS	France Telecom

Table 1: Contributors of the various modules of CLASSiC System 2: Appointment Scheduling

In the following sections, descriptions of each component will be given.

2 ASR

The ASR module uses the Cambridge HTK/ATK speech recogniser. It accepts live speech input and outputs N-best hypotheses with sentence-level confidence scores to the SLU component. A two-pass decoding (bi-gram language model decoding as the first pass and tri-gram rescoring as the second pass) scheme is used. The development of the French ASR system consists of phone set selection and dictionary construction, acoustic model training and language modelling.

A modified IPA based phone set is used for the French ASR system. There are altogether 38 phones (including sil/sp). This phone set has been used to build a French ASR system in Cambridge in the 1990s, which has shown good performance. The recognition dictionary is constructed manually, consisting of 514 words in the appointment scheduling domain.

The acoustic model training data consists of various sources as shown below:

- BREF-80: 9.9 hours of read French based on the *Le Monde* text source. The audio was recorded using close-talking microphones.
- MEDIA: 4.5 hours of French dialogue speech in the tourist information domain. The audio was collected under the French national project MEDIA (Automatic evaluation of man-machine dialogue

systems).

- FTREAD: 1.7 hours of read French by native speakers based on tourist information domain. The audio was recorded using the desktop based ATK dialogue system.
- FTVOIP: 2.4 hours of French dialogue in the appointment scheduling domain. The audio was recorded using the VoIP-based ATK dialogue system.

A narrow-band state-clustered triphone acoustic model was trained on these data. Since there are obvious channel differences between different corpora, corpus-based cepstral mean and variance normalisation was used. There are 2603 tied states and, on average, 4 Gaussian components per state.

The language model training data consists of various acoustic transcriptions and automatically generated texts. The transcriptions of BREF80, MEDIA corpus and some France Telecom collected MODEM fixing dialogues were used. Pseudo-data were generated from the pattern of semantic parser, in-domain appointment scheduling transcriptions and some hand-written in-domain texts. There are altogether 2.5M words in the text. Tri-gram language models were built for different corpora and then interpolated to yield the final tri-gram model.

3 Dialogue management

This section serves as a description of basic properties of the Appointment Scheduling (AS) system (Classic System 2) dialogue manager. First, the section defines the dialogue acts (semantics) to be used by the semantic parser, dialogue manager (DM), and user simulator. Then, it briefly describes the belief state monitoring, and dialogue policies used for the system. Finally, it details the general structure of an Appointment Scheduling dialogue.

3.1 Slot names and slot values

In the AS system, the following slots and values have been implemented:

- time = am, pm, outofhours, dontcare
- dayofweek = Monday, ..., Sunday, dontcare
- fromdayofweek = Monday, ..., Sunday, dontcare
- todayofweek = Monday, ..., Sunday, dontcare
- week = this, next, nextnext, 1, ..., 52, dontcare
- fromweek = this, next, nextnext, 1, ..., 52, dontcare
- toweek = this, next, nextnext, 1, ..., 52, dontcare
- dayofmonth = today, tomorrow, dayaftertomorrow, inthreedays, 1, ..., 31, dontcare
- fromdayofmonth = today, tomorrow, dayaftertomorrow, inthreedays, 1, ..., 31, dontcare
- todayofmonth = today, tomorrow, dayaftertomorrow, inthreedays, 1, ..., 31, dontcare

- month = this, next, nextnext, January,....,December, dontcare
- booking = option - refers to an offer of a specific appointment that is available.
- booking = final - refers to a unique appointment booking. Used only by the system to inform that the appointment has been booked or to confirm a booking.
- name = "1.1.2010 am" - is used only by system to inform about an agreed exact date. After confirming the date, the appointment is booked.
- name = none - is only used by the system to inform about no matching time slots being available.
- alt = x - is only used by the system to inform about an alternative appointment when it informs "name=none". Note that not always an alternative can be suggested.
- invalid = true - is only used by the system to inform that the provided constraints do not form a valid date, e.g., the 31st of February (dayofmonth=31, month=February) is not a valid date.
- weekr = this, next, nextnext - is only used by the system to provide relative information about the week value based on the current date.

The time slot: As the valid values for the slot "time" are only am, pm, outofhours, and dontcare, the dialogue manager converts any values from 9 to 12 to "am", any value from 13 to 17 to "pm", the rest is converted to "outofhours".

Relative slot values in the AS system: The dialogue manager allows input values like this, next, nextnext and today, tomorrow, dayaftertomorrow, inthreedays; however, these relative values are converted inside of the DM into absolute values, for example on the 1st of a particular month the following conversions happen - today => 1, tomorrow => 2, dayaftertomorrow => 3. It works similarly for "week=this" or "fromweek=next", etc.

3.2 Input dialogue act scheme

The AS system uses the following input (user – > system) dialogue acts:

- inform(a=x,b=y,...) - provide info
- inform(a=!x,b!=y,...) - provide info
 - e.g. "Anything but Italian" => inform(food!=Italian)
- request(a,b=x,c=y) - request info
- reqalts(a=x,b=y,..) - request alternatives within (or without) some constraints
- reqmore(a=x,b=y) - request more options
- confirm(a=x,b=y,..) - request a confirmation
- affirm() - yes
- negate() - no

- hello() - start
- silence() - silence
- thankyou() - "non-specific positive response from the user"
- ack() - back-channel ok
- bye() - bye
- hangup() - hangs up
- repeat() - request to repeat
- restart() - request to restart
- null() - null act
- deny (a=x) - no, a!=x
 - e.g. "I do not want a restaurant" => deny(type=restaurant)
- deny (a=x,b=y) - no, a!=x and give further information b=y
- inform(discourseAct=following) - inform the system that the user wants the first available appointment after the one which was offered.
- inform(discourseAct=previous) - inform the system that the user wants the first available appointment before the one which was offered.

In the BUDS system, the dialogue acts are internally broken down into one slot dialogue acts, e.g.:

- "How about tomorrow evening?" => inform(day=tomorrow,time=pm) is processed as inform(day=tomorrow); inform(time=pm)
- "Ok, let me recap. Monday the 21st of September in the afternoon. Do you confirm?" => confirm(day=Monday,date=21,month=September, time=pm) is processed as confirm(day=Monday); confirm(date=21); confirm(month=September); confirm(time=pm)

3.3 Output dialogue act scheme

The AS system uses the same dialogue acts as the Classic System 1 - TownInfo. However, it mainly uses the following output (system – > user) dialogue acts:

- request(a) - request value of slot a
- confirm(a=x) - confirm a=x
- select(a=x,a=x) - select between two values of one slot
- inform(a=x,b=y,...) - provide information about a and b

- `affirm(a=x,b=y)` - yes and give further info `a=x` and `b=y`
- `negate(a=x,b=y)` - no and give further info `a=x` and `b=y`
- `hello(name=x)` - start, the first available time slot is `x`
- `bye()` - bye

In addition, there are dialogue acts implementing specific appointment scheduling actions:

- the `inform(name=none,...)` dialogue act informs a user that there is no time slot given the previously specified constraints. E.g. "There are no available appointment slots on the date you provided."
- the `inform(name=none,invalid=true,...)` dialogue act informs a user that the previously specified constraints do not form a valid date. E.g. "Sorry, the 31st of February is not a valid date. Can you correct that?"
- the `inform(name=none,...,alt=x)` dialogue act informs a user that there is no time slot given the previously specified constraints, but there is a suitable alternative `x`. E.g. "There are no available appointment slots on the date you provided. However, I am available on `x`."
- the `inform(name=none,name!=x,name!=y,...)` dialogue act informs the user that there are no appointments except `x` and `y` given the constraints.
- the `inform(name=...,booking=option,...)` dialogue act offers an appointment to a user.
- the `inform(name=...,follows=x,booking=option)` dialogue offers an appointment with follows immediately after the appointment `x`. If the value for the name slot is "none" then there is no available appointment matching such constraints.
- the `inform(name=...,follows=x,booking=option)` dialogue offers an appointment with precedes the appointment `x`. If the value for the name slot is "none" then there is no available appointment matching such constraints.
- the `confirm(name=...,booking=final,...)` dialogue act confirms with a user whether the agreed appointment should be booked.
- the `inform(name=...,booking=final,...)` dialogue act means that an agreed appointment was saved into the database.

3.4 Belief state update and the summary actions

The DM implements the belief state as a dynamic Bayesian network using the BUDS framework [1]. The BUDS framework was also described in deliverable D1.4.2: "Summary-based policy optimisation" [2]. Using efficient approximations, a tractable Bayesian belief monitoring was achieved. For the Appointment Scheduling domain, two types of policies were implemented: a handcrafted and a stochastic trained policy. The stochastic policy is trained on a simulated user, using the Natural Actor Critic algorithm [1]. Both types of policies make use of summary actions [2].

The model parameters of the dynamic Bayesian network can be either handcrafted or trained using the Expectation Propagation algorithm [3] or the Natural Belief Critic algorithm [4]. The joint optimisation

of the model and the stochastic policy parameters can be performed by the Natural Actor and Belief Critic algorithm [5].

3.5 Simulated user

The parameters of the stochastic policy are trained from interactions with an agenda-based user simulator. This user simulator, which in the past has been used for testing, training and evaluating POMDP dialogue managers in the TownInfo domain, was extended for use in the Appointment Scheduling domain. In interaction with the AS system, the simulator tries to book an appointment on a time slot that is free on his own (randomly generated) user calendar. This time slot (or interval of time slots) is represented in terms of user goal constraints (in the form of slot value pairs) that the simulator then tries to satisfy in negotiation with the system. If the system informs the simulated user that no available time slots match the constraints specified, the user calendar is updated and a new user goal is constructed, representing the next free time slot on the user calendar. This process is repeated until a successful booking is made. For more details on the agenda-based user simulator, see deliverable D3.4 [6].

3.6 Structure of an appointment scheduling dialogue

In general, the DM allows a user to specify the time, day of week, week, month, and day of month independently of each other. Consequently, a user can provide a combination of a day of week and a day of month (or a day of month, a week, and a month) which is not valid according to the calendar. In this case, the DM informs the user that there is no appointment on these days: “inform(name=none, invalid=true, ...)” e.g. “There are no available appointment slots given the constraints ... you provided.” Furthermore, the DM requires the user to use either absolute values or intervals (from ..., to ...) for dayofweek and dayofmonth slots.

Once enough information is gathered so that no appointments are available or only one appointment is available, the DM informs the user by generating the act “inform(name=none, ...)”, i.e., “There is no available slot for an appointment given the constraints ...” or “inform(name=..., ...)”, i.e., “I have an available slot for an appointment on ... given the constraints ...”. The user can ask for an alternative, confirm slots, provide new information, reject, negate, deny or affirm the suggested date and time of the appointment.

If the date is affirmed, the system books the appointment and informs about the booking: “inform(name=..., booking=final)” e.g. “I have booked an appointment for you on 14.7.2010 in the morning.”. If a different dialogue act is used then the system acts accordingly. In case of dialogue acts such as “null(), silence(), thankyou(), and ack()”, the DM tries to confirm the suggested date and time: “confirm(name=..., booking=final)” e.g. “Is an appointment on 1.5.2010 in the morning OK with you?”. Also, if there is low confidence that the user acknowledged the offered appointment, then the booking is confirmed. If the user affirms with high confidence, the DM informs “inform(name=...,booking=final)” and the dialogue effectively ends.

4 Spoken Language Understanding

The UCAM Spoken Language Understanding (SLU) component is a handcrafted grammar for the Phoenix parser [7]. The parser returns the set of context-free sub-parses resulting in the largest word coverage, in

which non-terminals can represent various dialogue act types, item slots, item values, and intermediary symbols. It is important to note that the use of partial text fragments makes the parser robust to noisy input utterances. For example, the following grammar rules are used to decode ranges of days (? indicates optional rule expansions):

```

DAYRANGE -> FROMDAYOFWEEK TODAYOFWEEK
FROMDAYOFWEEK -> FROM DAY AT_TIME?
TODAYOFWEEK -> TO DAY AT_TIME?
FROM -> ( de ce? | du | a partir de ce? | a partir du )
TO -> ( jusque? au | jusque ce? | jusqu'a ce? )
AT_TIME -> ( a? | vers ) TIME
DAY -> ...
TIME -> ...

```

Each input ASR hypothesis is decoded individually, and identical semantic hypotheses are then merged by summing the inference evidence confidence scores of the corresponding ASR hypotheses. Let's consider the following input N-best ASR N-best list (log probabilities):

```

<asrhyp prob='-0.163676023'>      jeudi quinze heures      </asrhyp>
<asrhyp prob='-2.312538147'>      jeudi quinze              </asrhyp>
<asrhyp prob='-3.650744438'>      jeudi quinze heure       </asrhyp>
<asrhyp prob='-5.011305809'>      jeudi quinze juin        </asrhyp>
<asrhyp prob='-5.252517223'>      jeudi quinze heures     </asrhyp>
<asrhyp prob='-5.345973969'>      jeudi quinze a           </asrhyp>
<asrhyp prob='-6.132476330'>      jeudi jeudi quinze heures </asrhyp>
<asrhyp prob='-6.539419174'>      le quinze heures        </asrhyp>
<asrhyp prob='-6.769999504'>      jeudi et quinze heures  </asrhyp>
<asrhyp prob='-7.219739437'>      jeudi en quinze heures  </asrhyp>

```

Each candidate utterance in the list is decoded and merged to produce the following distribution over semantic hypotheses, capturing the uncertainty due to speech recognition errors as well as the lack of contextual information:

```

0.50: inform(dayofweek=Thursday,time="15")
0.30: inform(dayofweek=Thursday,dayofmonth="15")
0.10: inform(dayofweek=Thursday,dayofmonth="15",month=June)
0.09: inform(time="15")
0.01: inform(dayofweek=Thursday,totime="15")

```

5 Natural Language Generation

The natural language generation (NLG) component consists of a rule-based baseline system covering the full range of output system acts, and a reinforcement learning based component focusing on the generation of temporal referring expressions.

5.1 Handcrafted NLG baseline

UCAM has developed a baseline NLG component for the French AS domain using handcrafted language production rules. The rules associate each dialogue act with a string of text instantiating the values of each item. Here are the example production rules for the dialogue acts providing information about a time slot which was offered previously (? indicates optional items), e.g. `inform(time=am, dayofmonth=13, dayofweek=Thursday, week=next)`:

```
REALISATION -> Il s'agit d'un rendez-vous le DAYOFWEEK DAYOFMONTH? TIME? WEEK?
DAYOFWEEK -> jeudi
DAYOFMONTH -> 13
TIME -> durant la matinee
WEEK -> la semaine prochaine
```

Here are additional example realisations for various dialogue acts supported by the UCAM dialogue manager:

- `hello(name="2010-07-15 pm")`:
Bonjour. D'après les tests effectués sur votre ligne, un rendez-vous avec un technicien est nécessaire à votre domicile. Seriez-vous disponible le 15 juillet durant l'après-midi?
- `request(dayofweek)`:
Quels jours de la semaine êtes-vous disponible?
- `inform(name="2010-07-24 pm", dayofweek=Saturday, booking=option, week=next)`:
Un rendez-vous est possible le samedi 24 juillet durant l'après-midi, la semaine prochaine
- `inform(name=None, dayofweek=Thursday, dayofmonth="15", time=pm, month=July, alt="2010-07-16 pm")`:
Je suis désolé, mais personne n'est disponible le jeudi 15 juillet durant l'après-midi, cependant je peux vous proposer le 16 juillet durant l'après-midi
- `select(dayofmonth="4", dayofmonth="15")`:
Souhaitez-vous un rendez-vous le 4 ou le 15?
- `confirm(name="2010-07-16 pm", booking=final, dayofweek=Friday, week="28", time=pm, week=this)`:
Puis-je procéder à la réservation d'un rendez-vous pour le vendredi 16 juillet durant l'après-midi, cette semaine?
- `inform(name="2010-07-16 pm", booking=final, dayofweek=Friday, week="28", time=pm, week=this)`:
Un rendez-vous a été enregistré pour le vendredi 16 juillet durant l'après-midi, cette semaine. Vous pouvez maintenant raccrocher, ou demander à répéter les détails du rendez-vous.

5.2 Reinforcement learning based NLG

Reinforcement Learning (RL) techniques have been used to generate temporal referring expressions in the natural language generation module. Temporal referring expressions (TRE) are linguistic expressions that include date and time information and are used to refer to appointment slots. There are several ways that an appointment slot can be referred to. The following are a few examples:

- “12th January between 8am and 12am”
- “Tuesday 12th January between 8am and 12am”
- “Tomorrow in the morning”
- “This Tuesday at the same time”
- “12th in the morning”
- “Tuesday between 8 am and 12 am”
- “Tuesday this week between 8 am and 12 am”

As one can observe, the above expressions refer to the same appointment slot, but vary in terms of length, ambiguity, redundant information, user preference, and so on. Therefore an appointment scheduling system must know which format to use in its utterances to refer to appointment slots that it offers the user. These expressions contain two types of references: absolute references like “Tuesday”, “12th January”, and references relative to the user’s current date like “tomorrow”, “this Tuesday”, etc. Generating TREs therefore involves selecting appropriate pieces of information (like date, day, time, month, and week) to present and deciding how to present them (absolute or relative reference).

The objective of our NLG module is to convey a target appointment slot to the user using an expression that is easy to understand, but is not too long, and is preferred by the users. We address the issue of generating TREs by treating it as a statistical planning problem using Markov Decision Processes and Reinforcement Learning techniques, as also described by [8, 9].

Actions and States

Temporal referring expressions are divided into smaller units called *Temporal Referring Expression Units (TREU)*. They are then generated in a step by step manner by generating a sequence of TREUs that refer to the day (DY), date (DD), month (MM), week (WK) and time (TM) of the target appointment slot. For each of these units, the system must choose the format in which the corresponding unit information is to be presented: *absolute (abs)*, *relative (rel)*, or *none (nn)*. In addition to presenting information relative to the current date, it is also possible in some cases to present information relative to another slot that was already mentioned in the conversation. For example, “the 15th **at the same time**”. This format is called *relative to context (rc)*. For every TRE, the NLG module first generates a formal specification (e.g. $DY=abs, DD=rel, MM=nn, WK=abs, TM=rc$) which is then realised by a TRE realiser.

TREU	Choices
Day	absolute, relative, relative_context, none
Date	absolute, relative
Month	absolute, none
Week	absolute, relative, none
Time	absolute, relative_context

Table 2: TREU choices

Table 2 presents the choices that are available for each TREU. In total, there are 84 possible TREs that can be generated by the system (some combinations are ruled-out due to being clearly incomprehensible). The choices are made based on two factors: the distance between the current date and the target slot and the slot in context (a slot being morning or afternoon). Based on the distance, the target slot was classified to belong on one of four groups: G1 (1 to 2 slots away), G2 (3 to 6 slots), G3 (7 to 11 slots) or G4 (more than 11 slots). The slot in context represents whether there was any other slot already mentioned in the conversation so that the system has an option to use “relative_context” expressions to present day and time information. Information concerning the target slot’s group (i.e. distance to the target) and the slot in context make up the state space of the Markov Decision Process (MDP).

Data Collection

Data was collected using a web experiment. The experiment was divided into 2 tasks. In Task 1 of the experiment, users heard temporal referring expressions by playing audio files and selected one or more appointment slots that they thought the expression referred to on a customised calendar (see Figure 1). They are initially given today’s date so as to provide a context for inferring the meaning of relative components on the expression. In the second part of the experiment, users heard different expressions referring to a given appointment slot and were asked to rate them on a Likert scale of 1 to 6. This part of the experiment recorded users’ preferences. In contrast to Task 1, Task 2 was a dialogue split into 2 parts. Initially, the user is offered a date that they rate (see Figure 2), the user is then told that that date is rejected and the system offers them a second date that they rate (see Figure 3). The second date, therefore, can include relative to context expressions such as “the following day at the same time”.

During these trials, target slots were randomly selected from a two-week calendar. Several TREs were randomly generated to refer to the target slot and presented to users. When the users selected more than one appointment slot in Task 1, the TREs were assumed to be ambiguous to the users. Similarly in both Tasks 1 and 2, if users replayed the audio for some TREs and these TREs were also treated as ambiguous. The participants were given a chance to win 1 of 3 £50 Amazon vouchers. The number of participants for Task 1 was 73 and each participant had to classify 10 dates, so in total 730 TRE classifications were obtained. The number participants for Task 2 was 48, therefore obtaining preference ratings on 480 TRE audio files.

User simulation

The data collected in the web experiment was used to build a user simulation that responds to TREs generated by the system. The simulation responded with a dialogue action ($A_{u,t}$) to TREs based on the system’s dialogue act ($A_{s,t}$), system’s TRE ($TRE_{s,t}$), the distance group of the target slot (G), the previous slot in context (C), and the user’s calendar (Cal). The following probabilistic model was used to generate user dialogue actions.

$$P(A_{u,t}|A_{s,t}, TRE_{s,t}, G, C, Cal)$$

The user’s dialogue actions included accepting or rejecting a slot and requesting for clarification. The above model requested for clarification when the given TRE was deemed to be ambiguous. Otherwise, it either accepted or rejected it based on its calendar (which was set at the beginning of each dialogue).

[Return to the info page](#)

Appointment Scheduling Experiment: Task 1

You call up British Telecom to book an appointment for an engineer to come round to your house to fix your phone line.
Please play the audio which will give you an appointment slot (e.g. Tuesday between 2pm and 4pm).
Enter the letter of the slot in the calendar that the audio is referring to. For example, Tuesday 7th September between 2pm and 4pm is Slot C.
If it is not clear please enter more than one slot letter.

Today is Monday September 6th in the morning.

APPOINTMENT SLOTS: SEPTEMBER					
	Monday 6th	Tuesday 7th	Wednesday 8th	Thursday 9th	Friday 10th
AM	NOW	B	D	F	H
PM	A	C	E	G	I
	Monday 13th	Tuesday 14th	Wednesday 15th	Thursday 16th	Friday 17th
AM	J	L	N	P	R
PM	K	M	O	Q	S

Appointment date:

Slot letter:

Alternative slot letter (optional):

Alternative slot letter (optional):

Alternative slot letter (optional):

Alternative slot letter (optional):

Stage: Appointment Scheduling Part 1 Slide: 1 / 10

Figure 1: Screen shot of web interface: Task 1.

The screenshot shows a web browser window with the URL <http://www.macs.hw.ac.uk/InteractionLab/release/hhastie/RefExpressionsTask2.html>. The page title is "Appointment Scheduling Experiment: Task 2". The main content area contains the following text:

You will now be presented with 4 scenarios, each scenario contains two parts of dialogue where you are presented with an initial appointment slot and then an alternative slot. Please listen to the date phrases and rate your preference on a scale of 1-6. 1 is bad and 6 is great.

You must listen to ALL the audio and rate each one.

Today's date is Tuesday 7th September in the afternoon.

Dialogue Part 1

Operator: "We need to send out an engineer to your home. The first available appointment is:"

There are five rows, each with a "Play" button, a "Rating (1 is bad, 6 is great):" label, and a text input field. The ratings are: 5, 4, 4, 3, and 2.

Below the ratings is a "Next" button.

At the bottom of the page, there is a status bar that reads "Stage: Appointment Scheduling Slide: 1 / 8".

Figure 2: Screen shot of web interface: Task 2a.

The screenshot shows a web browser window with the URL <http://www.macs.hw.ac.uk/InteractionLab/release/hhastie/RefExpressionsTask2.html>. The page title is "Appointment Scheduling Experiment: Task 2". The main content area contains the following text:

Appointment Scheduling Experiment: Task 2

You will now be presented with 4 scenarios, each scenario contains two parts of dialogue where you are presented with an initial appointment slot and then an alternative slot. Please listen to the date phrases and rate your preference on a scale of 1-6. 1 is bad and 6 is great.

You must listen to ALL the audio and rate each one.

Today's date is Tuesday 7th September in the afternoon.

Dialogue Part 2

Operator: so you can't do Wednesday 8th September in the morning.

There are five rows of controls, each consisting of a "Play" button and a "Rating (1 is bad, 6 is great):" label followed by a text input field. The ratings are: 5, 4, 4, 2, and 1.

Below the ratings is a "Next" button.

At the bottom of the page, a status bar indicates "Stage: Appointment Scheduling" and "Slide: 2 / 8".

Figure 3: Screen shot of web interface: Task 2b.

Training

The learning agent was rewarded for each TRE that it generated. We used the following reward function.

$$\text{Reward} = UP * 10 + EL * 10 + CR * 10$$

Where UP is the user's preference (0 to 1), EL is the length of the TRE (0 to 1), CR is the user's response (1 if clarification request and 0 otherwise).

The Reward function penalises TREs that are long or ambiguous leading to user requests for clarification and rewards TREs that users prefer. User preference score (UP) for each TRE was obtained from Task 2 of the web experiment for data collection. Length of TREs (EL) was counted in TRE units (TREU) and is calculated against the maximum of 5 units (i.e. day, date, week, month, and time). Therefore, for an expression that has 3 units, EL is 3/5.

We trained the agent with the above user simulation model for 10000 runs using SARSA reinforcement learning algorithm to learn a TRE generation policy. During the training phase, the agent generated and presented TREs to the user simulation. When a dialogue begins, there is no appointment slot in context (i.e. $C = 0$). However, if the user rejects the first slot, the dialogue system state space sets C to 1 and presents the next slot. This is again reset at the beginning of the next dialogue. The agent was rewarded at the end of every turn based on the user's response, length of the TRE and user preference scores. It gradually explored all possible combinations of TREUs and identified those TREUs in different contexts that maximise the reward. Figure 4 shows the reward increasing during training.

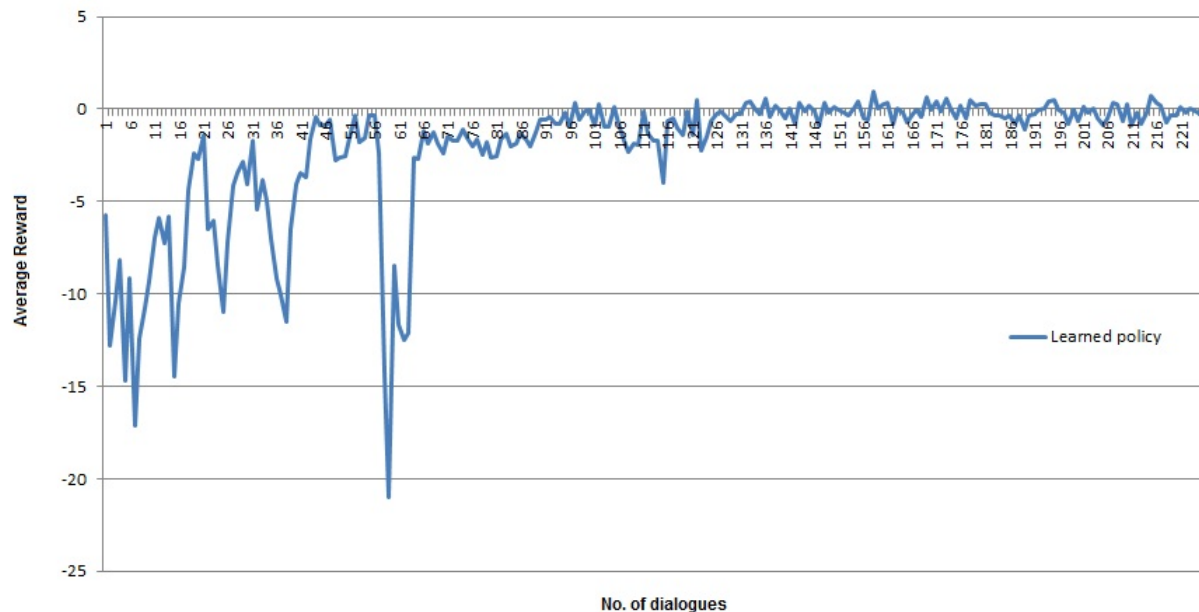


Figure 4: Learning curve

Distance Groups	Slot in context	No slot in context
G1	DY=rel;DD=abs;MM=nn;WK=nn;TM=abs	DY=rel;DD=abs;MM=nn;WK=nn;TM=abs
G2	DY=nn;DD=abs;MM=nn;WK=nn;TM=abs	DY=nn;DD=abs;MM=nn;WK=nn;TM=abs
G3	DY=nn;DD=abs;MM=nn;WK=nn;TM=abs	DY=nn;DD=abs;MM=nn;WK=nn;TM=abs
G4	DY=rel;DD=abs;MM=nn;WK=nn;TM=abs	DY=rel;DD=abs;MM=nn;WK=nn;TM=abs

Table 3: Learned policy

Table 3 presents the TRE generation policy learned by the agent. As one can observe, it used a minimum number of TREUs to avoid length penalties in the reward. In all cases, month and week information have not been presented at all. For target slots that were closest (in group G1) and the farthest (in group G4), it used relative forms of day (e.g. tomorrow, next Tuesday, etc.) This is probably because users dispreferred day information for in-between slots (e.g. day after tomorrow, day after day after tomorrow, etc).

Results

We evaluated the learned policy and three other hand-coded baseline TRE generation policies with our user simulation model. Each policy generated 1000 TREs in different states. The following are the baseline TRE generation policies:

1. **Absolute policy:** It always used absolute formats for all TREUs (i.e. DY=abs;DD=abs;MM=abs;WK=abs;TM=abs)
2. **Minimal policy:** It always used a minimal format with only date, month and time information in their absolute forms (i.e. DY=nn;DD=abs;MM=abs;WK=nn;TM=abs)
3. **Random policy:** It selected possible formats randomly for each TREU.

Table 4 presents the results of evaluation with simulated users. On average, the learned policy scores higher reward than all other baseline policies and the differences between the average reward of the learned policy and the other baselines are statistically significant ($p < 0.05$, two-tailed independent t-test). This shows that target slots can be presented using different TREs depending on how far they are from the current date and that such learned adaptation can produce less ambiguous, short and user preferred expressions.

TRE Policy	Average reward
Learned	-0.071 (± 3.75) *
Absolute	-4.084 (± 4.36)
Minimal	-1.340 (± 4.2)
Random	-8.21 (± 7.72)

Table 4: Evaluation with simulated users (* indicates $p < 0.05$)

Integration with System 2

The above trained policy has been integrated with System 2, using JNI to link the trained policy to the ATK-based system developed at Cambridge. The trained policy is activated at all points where the DM requires a Temporal Referring Expression to be generated, for example when offering an appointment slot to the user.

Conclusion

We have presented a principled method to generate temporal referring expressions (TREs) that refer to appointment slots in natural language utterances of the appointment scheduling dialogue system. We showed how the problem of generating TREs can be presented as a Markov Decision Process and can be solved using reinforcement learning techniques. We also presented a user simulation model that is sensitive to TRE units and how such a model can be populated using data from a web experiment. We showed that a TRE generation policy learned using our framework scores a significantly better reward than hand-coded policies.

6 TTS

The TTS component gets text from the NLG components and synthesises speech using the French Baratinoo expressive speech synthesiser. Baratinoo is the industrial speech synthesiser developed at France Telecom, using a unit-concatenation technique. For more details, refer to deliverable D5.2.1.

7 Software

The software package including all the previously described blocks is structured as follows:

- **atk**: general application toolkit
- **SemIO**: semantic decoder software used for word-level error model;
- **tHIS**: C++ sources of dialogue management and user simulation software, including:
 - *BUDSLib*: the BUDS dialogue manager;
 - * the source code related to the belief monitoring and the handcrafted and stochastic policy;
 - *HISLib*: the HIS dialogue manager;
 - *UMLib*: user simulation error modelling library;
 - *TDMan*: the test harness to run the simulator with one of the dialogue managers;
- **NLG**: the trained NLG component for Temporal Referring Expressions: java with JNI interface.
- **resources**: the domain ontology and database, configuration files, parameter files for user simulator and error model;

In the README files included in the package, more specific documentation is given on how to compile and run the system.

8 Conclusions

This document has described the prototype deliverable consisting of a French Appointment Scheduling system developed by Cambridge University, Edinburgh University, and Heriot-Watt University.

The speech recognition component uses the Cambridge HTK/ATK recogniser for which new models for French were developed. The POMDP dialogue manager uses the Bayesian Update of Dialogue State (BUDS) framework, for which an ontology defining the slots and possible values for the Appointment Scheduling (AS) domain was developed. For testing the dialogue manager, as well as training a dialogue management policy, the Cambridge agenda-based user simulator was extended to support the AS domain. The spoken language understanding component uses the Phoenix parser, for which a handcrafted grammar was written to enable the recognition of AS dialogue acts. The NLG component consists of a baseline system of handcrafted production rules and a more elaborate system for generating temporal referring expressions using reinforcement learning. Finally, the TTS component is provided by France Telecom and uses unit-concatenation.

This prototype will be evaluated on real users, along with two other AS systems developed by France Telecom. The results of these evaluations will be reported in deliverable D6.4.

Bibliography

- [1] B. Thomson and S. Young. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562 – 588, 2010.
- [2] F. Jurčiček, M. Gašić, B. Thomson, F. Lefèvre, and S. Young. Summary-based policy optimisation. Technical Report D1.4.2, CLASSiC, February 2010.
- [3] B. Thomson, F. Jurčiček, M. Gašić, F. Mairesse, K. Yu, and S. Young. Parameter learning for POMDP spoken dialogue models. *Proceedings of SLT*, 2010.
- [4] F. Jurčiček, B. Thomson, S. Keizer, F. Mairesse, M. Gasic, K. Yu, and S. Young. Natural Belief-Critic: a reinforcement algorithm for parameter estimation in statistical spoken dialogue systems. *Proceedings of INTERSPEECH*, 2010.
- [5] F. Jurčiček, B. Thomson, and S. Young. Natural Belief-Critic: a reinforcement algorithm for parameter estimation in statistical spoken dialogue systems (to appear). *ACM TSLP Special Issue on Machine Learning for Robust and Adaptive Spoken Dialogue Systems*, 2011.
- [6] S. Keizer, O. Pietquin, S. Rossignol, and S. Young. Agenda-based user simulations and EM training tools for Appointment Scheduling and TownInfo domains. Technical Report D3.4, CLASSiC, October 2010.
- [7] Wayne H. Ward. The Phoenix system: Understanding spontaneous speech. In *Proceedings of ICASSP*, 1991.
- [8] Srinivasan Janarthanam and Oliver Lemon. Learning to adapt to unknown users: Referring expression generation in spoken dialogue systems. In *Proceedings of ACL*, 2010.
- [9] Verena Rieser, Oliver Lemon, and Xingkun Liu. Optimising information presentation for spoken dialogue systems. In *Proceedings of ACL*, 2010.

The papers relevant to individual components are available for download on the CLASSiC website:
www.classic-project.org