ELSEVIER Interacting with Computers 14 (2002) 521–546

Interacting with Computers

www.elsevier.com/locate/intcom

Testing dialogue systems by means of automatic generation of conversations

R. López-Cózar*, A. De la Torre, J.C. Segura, A.J. Rubio, V. Sánchez

Dpto. Electrónica y Tecnología de Computadores, Universidad de Granada, 18071 Granada, España, Spain

Received 17 April 2001; revised 23 November 2001; accepted 15 March 2002

Abstract

This paper presents a novel technique that allows testing spoken dialogue systems by means of an automatic generation of conversations. The technique permits to easily test spoken dialogue systems under a variety of lab-simulated conditions, as it is easy to vary or change the utterance corpus used to check the performance of the system. The technique is based on the use of a module called *user simulator* whose purpose is to behave as real users when they interact with dialogue systems. The behaviour of the simulator is decided by means of diverse scenarios that represent the goals of the users. The simulator aim is to achieve the goals set in the scenarios during the interaction with the dialogue system. We have applied the technique to test a dialogue system developed in our lab. The test has been carried out considering different levels of white and babble noise as well as a VTS noise compensation technique. The results prove that the dialogue system performance is worse under the babble noise conditions. The VTS technique has been effective when dealing with noisy utterances and has lead to better experimental results, particularly for the white noise. The technique has permitted to detect problems in the dialogue strategies employed to handle confirmation turns and recognition errors, suggesting that these strategies must be improved. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Dialogue system; Dialogue management; Speech recognition; Sentence understanding; Natural language generation; Speech synthesis; User simulator

1. Executive summary

This paper is organised in five sections. Section 2 presents an introduction to the state-of-the-art of the spoken dialogue system technology. It mentions some example of systems

* Corresponding author. Tel.: +34-958-243271; fax: +34-958-243230.
E-mail addresses: rlopezc@ugr.es (R. López-Cózar), atv@ugr.es (A. De la Torre), segura@ugr.es (J.C. Segura), rubio@ugr.es (A.J. Rubio), victoria@ugr.es (V. Sánchez).

and focuses on some of the problems concerned with the design, development and performance. Section 3 presents several user simulation techniques that can be found in literature and describes the main differences between these techniques and the technique proposed in this paper. Section 4 presents the technique proposed in this paper. Initially, this section describes briefly the dialogue system used in the experiments and indicates how it has been interconnected to a user simulator. Later, this section indicates how to build a simulator for any spoken dialogue system, focusing on the corpora need for the set up. This section also presents an algorithm explaining how the simulator responses can be generated. The section ends with an example of a generated dialogue between the simulator and the dialogue system used in the experiments. Section 5 presents the experimental results. It includes the main features of the recognition set up and describes the scenario and utterance corpora used. The evaluation is focused on word accuracy, sentence understanding and task completion. Later, the section includes a discussion on the results obtained. The section finishes explaining how the technique proposed in this paper permitted to detect flaws of the dialogue system. Section 6 presents some limitations of the technique. Finally, Section 7 presents the conclusions and Section 8 indicates possibilities for future work.

2. Introduction

Speech is the most natural, flexible and comfortable method for the communication among humans. For many years researchers have tried to develop machines able to communicate with human beings using speech. Recent advances in speech recognition, speech understanding and speech synthesis have made possible to build systems able to allow a speech-based communication in restricted domains. These systems are called *spoken dialogue systems* or *conversational interfaces*, and aim to be 'intelligent' speech-based interfaces. Their goal is allowing users to talk naturally, like if they were talking to a human operator who provides some information or service. Many spoken dialogue systems for a variety of applications such as air travel information (Zue et al., 1994), weather forecast (Zue et al., 2000), automatic call routing (Lee et al., 2000), language learning (Ehsani et al., 2000), information retrieval (Fedrico, 2000) and railway ticket reservation (Lamel et al., 2000) have been developed.

These systems are difficult to be specified, designed, developed and maintained. Their development requires expertise in multiple domains, mainly, speech recognition, natural language understanding, dialogue management, sentence generation and speech synthesis. Because of these problems, among others, most currently developed dialogue systems are restricted to specific domains. Most of them are based on many application-specific collected examples and are specialised to perform determined tasks. Using the domain knowledge, the perplexity of the task to be performed by the dialogue system can be reduced and pretty good results can be obtained. However, this strategy leads to specialised systems for specific tasks. If the application domain changes, it is usually fairly time consuming to develop a new system for the new domain, although some components from the developed system could be reused. Some attempts to develop systems able to operate in multiple domains simultaneously have been made (Lin et al., 1999). The development of spoken dialogue systems requires the collection and labelling of large sets of user–system interactions. Among other considerations, these sets allow to represent the phonemes by acoustic models. Each user has its own way to pronounce words, and even a same user will not pronounce twice the same way the same utterance. So, a large amount of training data must be collected in order to provide speaker-independent speech recognition (Pfau et al., 1999; Brieussel-Pousse and Perennou, 1999). However, the training data can be difficult to obtain, and the transcription of real utterances can be an expensive and time-consuming task.

The increasing commercial use and sophistication of spoken dialogue systems has originated the need to develop tools to facilitate the creation and test (Gibbon et al., 2000; Bernsen et al., 1999; Edgington et al., 1999; Müller and Schröder, 1999; McTear, 1999; Pargellis et al., 1999). The *Wizard of Oz* (WOZ) technique has been widely used in the initial stages of development to check the performance (Ammicht et al., 1999; Ehrlich, 1999). When it is used a human operator decides the responses of the dialogue system and the users are made to believe that they are actually talking to a machine. The operator receives either the user utterance or the recogniser output and controls the conversation with the user. Testing new strategies using this technique requires extensive interactions with users. If the dialogue system is already performing for the public, the interaction is available. However, in initial development stages, extensive experimentation with real users is not always possible. In most occasions, the systems are just tested by the developers themselves. Testing a dialogue system can be a very costly and time-consuming task, as every time a new strategy is tested, new speech data must be collected.

3. Techniques for user simulation

The idea of using a simulation technique to check the performance of spoken dialogue systems is not a new one. For example, McAllaster and Gillick (1999) present a technique to test speech recognition algorithms using simulated speech data. In this paper, the key methodological tool is the ability to simulate speech data that correspond to a particular text from a set of acoustic models.

Levin et al. (2000) present a user simulator for learning dialogue strategies. This user is defined as a generative stochastic model that, given the dialogue system current state and a prompt, produces the semantic representation of an utterance in a template form similarly to a template generated by the user. In this work recognition and understanding errors are not considered.

Polifroni et al. (1998) present a work where simulated interactions are used to create log files to evaluate the JUPITER system, designed to provide weather information. Polifroni and Seneff (2000) present a modular architecture for the Galaxy-II dialogue system were several components are interconnected by a hub. One of these components is a ‘batchmode’ server that stands in place of a user interface as is able to extract appropriate inputs from a log file to initiates dialogue turns.

In Scheffler and Young (2000) the interaction between the system and the user simulator takes place solely at the intention level, rather than the word or the speech signal level. An intention represents the actual information (concepts) that a dialogue participant wants to convey. Each user utterance is viewed as a sequence of intentions.

Eckert et al. (1997, 1998) propose a bigram user model that specifies the probability of a possible user utterance conditioned only on the preceding system utterance. The bigram user model takes into account neither the history of the dialogue nor the aims of the user. So this approach is not valid for dialogues where different user utterances are interdependent or where confirmation subdialogues are necessary.

3.1. Main differences between the proposed and other user simulation techniques

A major difference between the technique proposed in this paper and other user simulation techniques is that the former considers phenomena existing at the word or signal level, as the simulator communicates with the dialogue system using utterances (voice signal files) recorded by several speakers. Additionally, it covers speech generation. In most previously developed techniques, the simulator communicates with the dialogue system using only the semantic representations of utterances, being word or signal level phenomena ignored, and speech generation is not considered.

There is also a difference concerned with how the user is modelled. For example, Eckert et al. (1997, 1998) show that the user model takes into account neither the dialogue history nor the aims of the real user. In Scheffler and Young (2000) and in the method proposed in this paper these aspects are taken into account. In (Scheffler and Young, 2000) a probabilistic model of user behaviour is used when the user has more than one option that is consistent with the current goal. The technique proposed in this paper uses a deterministic model of user behaviour that uses a set of rules to associate prompts generated by the dialogue system to utterance types the simulator must generate.

Other difference is in the way that errors are modelled. The proposed technique takes into account the real errors generated by the speech recogniser and the linguistic analyser. In (Scheffler and Young, 2000) speech recognition and understanding errors are modelled by grouping the different error types together and considering only the total distortion that takes place between the original intention in the mind of the user and the eventual representation obtained by the system. An error model is defined to generate valid substitutions of these intentions.

The utterances the simulator uses to interact with the dialogue system represent also a difference. In (Scheffler and Young, 2000) utterances are created by a procedure that adds and substitutes intentions. Once a 'correct' intention has been added to an utterance, the error model is used to generate valid substitutions of these intentions. The substitutions are restricted to groups of intentions called *intention groups*. These groups are chosen to correspond with the intentions allowed at specific points in the syntax. In the technique proposed in this paper, the simulator interacts with the dialogue system using real utterances (voice signal files) whose semantic representations are included in scenarios representing user goals. Considering the semantic meaning, a procedure for utterance selection determines the utterance, the simulator must use at every moment to answer the prompts of the dialogue system.

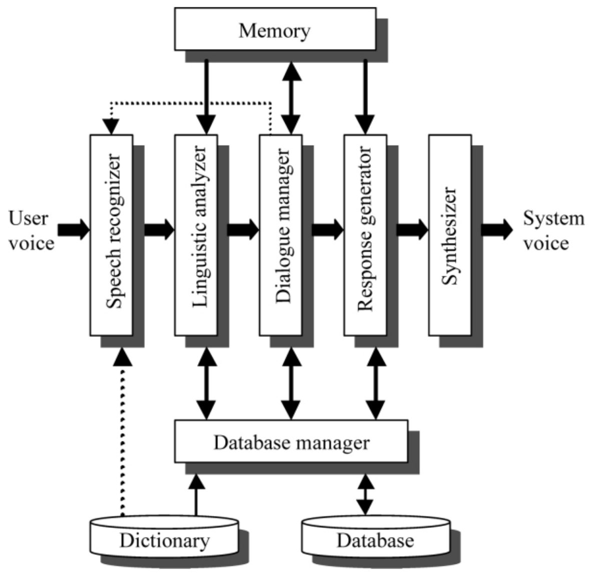

Fig. 1. Structure of the SAPLEN dialogue system.

4. The proposed ADG technique to test spoken dialogue systems

This section describes generally how to apply the ADG technique to test the performance of any spoken dialogue system. Although the paper focuses on the SAPLEN system (López-Cózar et al., 1999, 2000a,b) as it has been used in the experiments, the technique is general and could be applied to test any spoken dialogue system. In addition to the examples concerning the SAPLEN system, this section also indicates how the technique could be applied to test a dialogue system designed for the ATIS domain.

Fig. 1 shows the structure of the SAPLEN system. This system has been previously developed in our lab to deal with the product orders and queries of fast food restaurants' clients using the telephone. The system has the typical structure of current spoken dialogue systems. It consists of a speech recogniser, a linguistic analyser, a dialogue manager, a response generator and a synthesiser. The speech recogniser converts user utterances to text strings. The linguistic analyser converts the text string into conceptual representations that capture the semantic meaning of the recogniser outputs. The dialogue manager is the core of the system. It decides the next action to be taken by the system as well as the next response to be generated. The response generator builds this response in text format. Finally, the synthesiser transforms the text response into synthesised voice. In the experiment we have used the ADG technique to test the performance of this system.

Fig. 2 presents the connection between a user simulator and the SAPLEN.

As can be observed, the simulator generates its responses considering the semantic frames (Allen, 1995) created by the dialogue system as well as its prompts. The frames are created by the dialogue system in real time from the utterances (voice signal files) it receives as inputs from the simulator. Therefore the frames can be affected by recognition

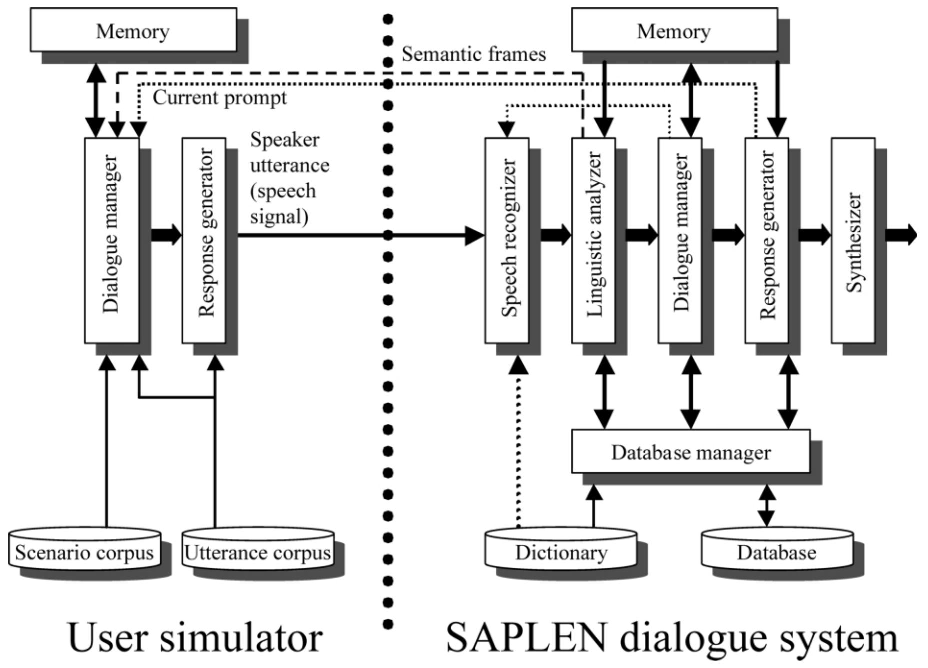

Fig. 2. Connection between a user simulator and the SAPLEN dialogue system.

and understanding errors that must be handled by the dialogue system. The goal of the ADG technique is testing the dialogue system considering that the semantic representations it obtains from the analysis of the utterances can be affected by these errors.

4.1. Scenario corpus

The ADG technique is based on the use of a scenario corpus that represents typical goals of users. In every dialogue, the simulator interacts considering a particular scenario. The scenarios follow the so-called *plan-based* modelling scheme. They are based on the assumption that users generally have goals and plans in mind when they interact with dialogue systems. So the purpose of such systems is to recognise the user goals and perform the corresponding actions. To set up the technique a wide range of scenarios must be carefully designed considering a representative sample of possible user goals. The goals can be represented in the scenarios semantically, using frames for example. This way, the simulator can easily extract the data items in the goals to easily answer the dialogue system prompts for missing data in partially understood utterances. In order to represent the goals in the scenarios we have used the semantic representations previously created to represent the utterances. Fig. 3 shows a sample scenario used in the experiments.

```
# the user wants to order for a ham sandwich             # the user phone number

AMOUNT="1"                                               PHONE_NUMBER="958275360"
FOOD_TYPE="sandwich"
INGREDIENT="ham"

# the user wants to order for a big beer                 # the user zip code

AMOUNT="1"                                               ZIP_CODE="18001"
DRINK="beer"
SIZE="big"

# the user wants to order for a vegetal and turkey salad # the user address

AMOUNT="1"                                               ID_ADDRESS="street"
FOOD_TYPE="salad"                                        NAME="andalucia"
INGREDIENT="vegetal and turkey"                          BUILDING_NUMBER="58"
                                                         FLOOR="first"
                                                         LETTER="e"
```

Fig. 3. A sample scenario.

As can be observed, the scenario defines the user goals and the data needed by the delivery service. The SAPLEN system may ask for this data during the interaction with the simulator.

4.2. Utterance corpus

The utterance corpus contains all the possible utterances the simulator may need to interact with the dialogue system. The utterances should be recorded by several speakers in order to test the recogniser and should be widely representative of typical utterances in the domain. They can be recorded in *clean* conditions and different degrees and noise types can be added latter artificially to obtain experimental results under different lab-simulated, noise conditions. If an utterance U conveys n data items d_i, then the data items are n new utterances that need to be added to the corpus. For example, let us suppose that in the ATIS domain, the utterance "*I want to travel from Boston to Chicago*" is in the corpus. This utterance can be seen as conveying two data items: SOURCE_CITY = "Boston" and DESTINATION_CITY = "Chicago". Then, the utterances "*Boston*" and "*Chicago*" should be added to the utterance corpus. Similarly, let us suppose that in the fast food domain, the utterance "*I want two ham sandwiches*" is in the corpus. This utterance can be seen as conveying three data items: AMOUNT = "two", FOOD_TYPE = "sandwich" and INGREDIENT = "ham". Then, the utterances "*two*", "*sandwich*" and "*ham*" should be added to the utterance corpus.

It is necessary to define a procedure that permits the simulator use the utterances properly to generate its responses. The ADG technique proposes to create a phonetic transcription and a semantic representation for every utterance in the corpus. This way when the simulator needs an utterance to generate a response, it can select one considering its semantic representation. This way, the phonetic transcriptions can be used to create conversation log files to check the system a posteriori and the semantic representations can be used for the utterance selection. As different utterances may share the same semantic representation, i.e. the same meaning, the simulator can generate different conversations even for the same scenario. For example, let us suppose the simulator used in the experiments needs to select an utterance whose semantic representation is (AMOUNT = "1", FOOD_TYPE = "sandwich", INGREDIENT = "ham"). Then, it could choose any of the following utterances: "*I would like to have a ham sandwich*", "*A ham sandwich, please*", "*Please, a ham sandwich*", etc. as they all share the required meaning.

4.3. Definition of actions

The ADG technique is also based on a set of *if…then…else* rules that the simulator uses to know the actions it must carry out to answer every possible prompt generated by the dialogue system. In order to define these rules for the dialogue system, it is necessary to examine the behaviour of the system a priori to know all the possible prompts it can generate. These actions depend on the task to be performed by the dialogue system. For example, in the case of the ATIS domain, a typical user goal may be to make a flight reservation. Then, a simulator for this domain must be able to make flight reservations. In addition to the domain-dependent actions two action types must be specially considered, as they are domain-independent; these actions are *error recovery* and *confirmations*.

4.3.1. Error recovery and confirmations

If frames are used to represent the meaning of the recogniser outputs, recognition errors can make some slots of the semantic frames to be empty as a consequence of deletion errors. In order to recover from these errors, dialogue systems generally prompt users to repeat the missing data. To take this procedure into account, the simulator uses the goals (frames) set in the scenarios as well as their corresponding data items (slots) to solve the dialogue system correction attempts. This recovery strategy, concerning the fast food domain, is illustrated in the Example 1.

Speaker	**Utterance**	**Simulator action**
SYSTEM:	What would you like to have?	-
SIMULATOR:	I want one ham sandwich	Use a scenario goal
SYSTEM:	How many sandwiches did you say?	-
SIMULATOR:	One	Correction
SYSTEM:	Ok, one ham sandwich. Please, say your phone number	-

In other occasions, word-change errors may be unnoticed by dialogue systems. All the expected slots in frames may be filled but the errors may provoke wrong semantic representations. For example, in the ATIS domain, a user may utter "*I want to travel from Boston to Los Angeles*" and the system may understand that the user wants to travel from Boston to Dallas. The user will be asked to confirm both cities if the system uses an explicit confirmation strategy (Bouwman and Hulstijn, 1998; Krahmer et al., 1999; Souvignier et al., 2000; Niimi et al., 2000; Meng et al., 2000; McTear et al., 2000). If it uses an implicit confirmation strategy, both cities will be mentioned in the system next turn and they will be considered confirmed unless the user makes a correction. Both confirmation strategies can be easily set up using the technique proposed in this paper. In the case of explicit confirmations, the simulator can compare the semantic representation obtained by the analysis of the recogniser output against the utterance correct semantic representation. If both coincide then the simulator can generate a positive confirmation and it can generate a negative confirmation otherwise. The Example 2 shows how the simulator built to test the SAPLEN system is able to detect and repair understanding errors. In this example, the dialogue system uses an implicit confirmation strategy to obtain the confirmation for the product order.

Speaker	Utterance	Simulator action
SYSTEM:	What would you like to have?	-
SIMULATOR:	I want one ham sandwich	Use a scenario goal
SYSTEM:	Ok, one cheese sandwich. Please, say your phone number	-
SIMULATOR:	it is wrong	Correction
SYSTEM:	Ok, I remove the cheese sandwich. Please, repeat your last order	-
SIMULATOR:	I want one ham sandwich	Use a scenario goal
SYSTEM:	Ok, one ham sandwich. Please, say your phone number again	-

In order to handle the comparison of semantic representations, two types of wrong semantic representation can be considered: *partial* and *contradictory*. A semantic representation obtained by a dialogue system $\mathrm{SR}_{\mathrm{obt}} = \{s_1, s_2, \ldots, s_n\}$ is considered partial with regard to the correct semantic representation $\mathrm{SR}_{\mathrm{cor}} = \{s'_1, s'_2, \ldots, s'_m\}$ if $\mathrm{SR}_{\mathrm{obt}}$ only contains some of the slots s_i in $\mathrm{SR}_{\mathrm{cor}}$ and it does not contain any slot that is not in $\mathrm{SR}_{\mathrm{cor}}$, i.e. $\mathrm{SR}_{\mathrm{obt}} \subset \mathrm{SR}_{\mathrm{cor}}$. On the other hand, $\mathrm{SR}_{\mathrm{obt}}$ is contradictory with regard to $\mathrm{SR}_{\mathrm{cor}}$, if $\mathrm{SR}_{\mathrm{obt}}$ contains at least one slot that is not in $\mathrm{SR}_{\mathrm{cor}}$, i.e. $\mathrm{SR}_{\mathrm{obt}} \not\subset \mathrm{SR}_{\mathrm{cor}}$. We consider that a partial semantic representation provokes a partial semantic error and a contradictory semantic representation provokes a contradictory semantic error.

Let us consider that a dialogue system uses an implicit confirmation strategy and that it prompts for the missing data caused by deletion errors. If a partial understanding error occurs then the simulator does not need to initiate a correction turn, as the dialogue system will try to correct the error by prompting for the missing data. However, if a contradictory understanding error occurs, then the simulator must initiate a repair subdialogue, as it cannot accept the obtained wrong semantic representation. In a conversation between a real user and a dialogue system, it is the user responsibility to notice the error and start a subdialogue to repair it. In order to model this procedure, the simulator can compare the semantic representations and can initiate a repair subdialogue if a contradictory understanding error occurs. Notice, however, that if the system uses an explicit confirmation strategy, then the simulator must not initiate a repair subdialogue in case of error. Instead, it must wait for the dialogue system confirmation prompt and then must generate a negative confirmation, as can be seen in the Example 3.

Speaker	**Utterance**	**Simulator action**
SYSTEM:	Please, say your phone number	-
SIMULATOR:	9 5 8 1 2 3 1 4 7	Use a scenario goal
SYSTEM:	Did you say 9 5 8 1 4 3 1 4 6?	-
SIMULATOR:	No	Correction
SYSTEM:	Ok, please, repeat your phone number	-

4.3.2. Scenario goals management

When the simulator selects a goal to answer a dialogue system prompt, the goal can be analysed to extract its data items. These items let the simulator generate the appropriate answers in case the dialogue system does not understand completely the previous simulator utterance. For example, in the ATIS domain, the utterance "*I want to travel from Boston to Chicago next Monday*" can be represented as the goal shown in the Example 4.

```
GOAL_TYPE="reservation"
SOURCE_CITY="Boston"
DESTINATION_CITY="Chicago"
DATE="next Monday"
```

This goal can be considered as containing four data items: GOAL_TYPE = "reservation", SOURCE_CITY = "Boston", DESTINATION_CITY = "Chicago" and DATE = "next Monday". For example, a simulator could use these items to answer the dialogue system prompt: "*Please, repeat the destination city*". Of course, at least an utterance for every goal in the scenarios and every data item must be previously recorded. A simulator response can be obtained by three different ways. Firstly, looking for an appropriate goal in the scenario. For example, this way allows the simulator to answer the dialogue system prompt: "*What can I do for you*?" with the response: "*I want to travel from Boston to Chicago*". Secondly, analysing the state of the goals in the scenario and considering which of them have been achieved already. In this case, the simulator answer is not included in the scenario, but it is derived from the conversation status. For example, this way lets the simulator answer the dialogue system prompt: "*Can I do anything else for you*?" with the response: "*No, thanks*". Finally, a third way to generate a response is based on a dialogue history analysis. Again, the simulator response is not explicitly set in the scenario; it is decided considering the information the dialogue system has understood during the interaction. For example, this way permits the simulator answer the confirmation prompt: "*Did you say you want to travel from Boston to Chicago*?" with the response: "*Yes*".

4.4. ADG algorithm

To start the dialogue generation the simulator selects a scenario and analyses it to know the goals it contains. The simulator uses 31 rules that define correspondences between dialogue system prompts (for example, *prompt_for_product_order*, *prompt_for_zip_code*, etc.) and possible goals in the scenario (for example, *food_order*, *user_zip_code*, etc.). For every prompt the simulator tries to find a non-previously used goal in the scenario that is associated to the prompt considering the correspondence rules. If a goal is found, then it is marked as *used* and its semantic representation (frame) is used to select an utterance from the utterance corpus. The semantic representation is analysed to detect the data items (slots) it contains, in order to prepare the simulator answers for possible error recovery prompts. If no unused goals associated to the prompt are found then the simulator analyses the history to decide its next response. For example, if the prompt is "*Would you like to have anything else*?" and all the orders in the scenario have been correctly understood by the system then the simulator uses a rules that indicate to generate a negative response (for example, "*no thanks*"). The next algorithm shows the procedures explained in this section to generate the simulator responses.

```
simulator_response_generation(Inputs: prompt, $SR_{obt}$; Output: voice_signal_file)
  /* section 1: implicit confirmations */
  if (prompt is not a explicit confirmation) and (not first response)
     {if ($SR_{obt} \neq SR_{cor}$) and ($SR_{obt} \not\subset SR_{cor}$) then SR = ERROR_INDICATION
     return() }
  else
  /* section 2: explicit confirmations */
  if (prompt = TELEPHONE_OK?) then { if ($SR_{obt} = SR_{cor}$) then SR =
  AFFIRMAT_CONF
     else SR = NEGAT_CONF }
     else
     if (prompt = ZIP_CODE_OK?) then { if ($SR_{obt} = SR_{cor}$) then SR =
     AFFIRMAT_CONF
        else SR = NEGAT_CONF }
     else
     …
     if (prompt = ORDERS_OK?) then { check_ordered_products()
     if (ok) then SR = AFFIRMAT_CONF
     else SR = NEGAT_CONF }
  else
  /* section 3: product-order error recovery */
  if (prompt = AMOUNT?) then SR = AMOUNT_ITEM
  else
  if (prompt = SIZE?) then SR = SIZE_ITEM
  else
  …
  if (prompt = TASTE?) then SR = TASTE_ITEM
  else
  /* section 4: scenario goals' management */
```

```
    **if** (prompt = SOMETHING_TO_EAT?) **then** { search_scenario_for_food_order(goal)
        SR = goal
        create_itmes(goal) }
    **else**
    **if** (prompt = SOMETHING_TO_DRINK?) **then** { search_scenario_for_drink_order(goal)
        SR = goal
        create_items(goal) }
    **else**
    …
    **if** (prompt = ANYTHING_ELSE?) **then** { search_scenario_for_order(goal)
        SR = goal
        create_items(goal) }
    /* section 5: response generation */
    look_for_file(SR, file.txt, file.samples)
    include_in_dialogue_file(prompt, file.txt)
    voice_signal_file = file.samples
    /* section 6: dialogue history storage */
    store_in_memory($SR_{obt}$)
    /* set correct semantic representation as the desired semantic representation */
    $SR_{cor}$ = SR
**End**
```

All the generated dialogues are stored in files that are written by both the simulator and the dialogue system during the interaction. The simulator writes the phonetic transcriptions of the utterances it uses to generate its responses and the dialogue system writes the recogniser outputs. The simulator also writes an understanding tag if the semantic representation obtained by the dialogue system coincides with the correct semantic representation. At the end of the dialogue the simulator checks whether all the goals in the scenario were correctly understood by the dialogue system (considering all the semantic representations obtained by the dialogue system) to write a task completion tag. In order to automatically test the performance of the dialogue system we wrote a program that takes as input the generated dialogues and obtains the figures of the evaluation metrics considered.

4.5. Example of a generated dialogue

To finish Section 4, the Example 5 shows a sample of a generated dialogue including the recogniser outputs. The dialogue has been translated from Spanish into English. The example deliberately includes some *forced* recognition errors to show the simulator error recovery capability. In order to force the errors, we recorded some sentences pronouncing some words deliberately bad to confuse the recogniser and then made the simulator use these sentences. We created manually the correct phonetic transcriptions and semantic representations for these sentences, so the simulator had no problem to select and use them as if they were correct. These sentences were used by the simulator in the turns (2), (7) and (8).

SAPLEN: Good evening, welcome to our restaurant. This is our telephone service automatic system. We inform you that the least price for the home delivery service is one thousand pesetas. What would you like to have?

(1) **SIMULATOR:** one ham sandwich
Recogniser output: one ham sandwich
SAPLEN: ok, one ham sandwich. Please, say your phone number.

(2) **SIMULATOR:** 9 5 8 2 7 5 3 6 0
Recogniser output: 9 5 8 2 7 5 6 6 0
SAPLEN: ok, I need to confirm your phone number. Did you say 9, 5, 8, 2, 7, 5, 6, 6, 0?

(3) **SIMULATOR:** no
Recogniser output: no
SAPLEN: ok, Please, say your phone number again.

(4) **SIMULATOR:** 9 5 8 2 7 5 3 6 0
Recogniser output: 9 5 8 2 7 5 3 6 0
SAPLEN: ok, I need to confirm your phone number again. Did you say 9, 5, 8, 2, 7, 5, 3, 6, 0?

(5) **SIMULATOR:** yes
Recogniser output: yes
SAPLEN: ok, I need to confirm your address. I think it is Andalucía avenue, number 58, third floor, letter e. Is it ok?

(6) **SIMULATOR:** yes
Recogniser output: yes
SAPLEN: ok. Would you like to drink something?

(7) **SIMULATOR:** a big beer
Recogniser output: a beer
SAPLEN: ok, a beer. Which size would you like?

(8) **SIMULATOR:** big
Recogniser output: normal
SAPLEN: ok, one normal beer. Would you like anything else?

(9) **SIMULATOR:** it is wrong
Recogniser output: it is wrong
SAPLEN: ok, I remove the normal beer. Please, repeat your last order

(10) **SIMULATOR:** a big beer
Recogniser output: a big beer
SAPLEN: ok, a big beer. Would you like anything else?

(11) **SIMULATOR:** a vegetal and turkey salad
Recogniser output: a vegetal and turkey salad
SAPLEN: ok, a vegetal and turkey salad. Anything else?

(12) **SIMULATOR:** no
Recogniser output: no
SAPLEN: ok, I need to confirm your orders. I think you ordered for a ham sandwich, a big beer and a vegetal and turkey salad. Is it ok?

(13) **SIMULATOR:** yes
Recogniser output: yes
SAPLEN: The price is 1050 pesetas. Is it ok?

(14) **SIMULATOR:** yes
Recogniser output: yes
SAPLEN: ok. The delivery service will take 30 minutes approximately to get to your address. Is it ok?

(15) **SIMULATOR:** yes
Recogniser output: yes
SAPLEN: ok. Thank you very much for using our telephone service automatic system. Good evening.

As can be observed in the example, in addition to the turns really employed for product ordering, the user must employ several dialogue turns to inform and confirm the delivery service data (telephone number, zip code and address). The SAPLEN system uses the phone number to query a database and decide whether the user is already known. If the user is known, the system just asks to confirm the address. Otherwise, the system asks for the zip code and afterwards for the address. At the end of the dialogue the user is asked to confirm the ordered products, the price to pay and the estimated delivery service time. Although our aim is to generate simulated dialogues as real as possible, there are some limitations in the current set up of the ADG technique that makes simulated dialogues to be different from the real ones. These limitations are described in Section 6.

5. Experiments

The aim of the experiments is testing the SAPLEN system using the ADG technique to obtain a performance estimation under noisy conditions using a VTS noise compensation technique (Moreno, 1996; Stern et al., 1997).

5.1. Speech recognition set up

The SAPLEN system uses an HTK-based recogniser (Hain et al., 1999; Woodland et al., 1999) that receives as input the user utterance and a bigram determined by the dialogue manager (Albasano et al., 1997; Nasr et al., 1999; Siu and Ostendorf, 2000). The dialogue system can use up to 126 different bigrams that have been previously compiled from an utterance corpus concerning the fast food domain. Among them, 109 bigrams can be used for the recognition of the user data (phone number, zip code and address) and the 17 remaining bigrams can be used for the recognition of product orders, queries, confirmations and corrections of the users. The speech recognition is based on word classes to reduce the size of the language models (Smadli et al., 1999; Siu and Ostendorf, 2000; Niesler and Woodland, 1999). A word class is a set of words that have syntactic similarities (e.g. sizes, tastes, etc.). The vocabulary size is about 2000 words, including restaurant-product names, names of streets, avenues, squares, etc. The recogniser uses acoustic models trained with the 4767 sentences of the Albayzín geographical database (Casacuberta et al., 1991). This database was decimated at 8 KHz. During the experiments, the recogniser had to perform under different lab-simulated noisy conditions, with and without a VTS compensation technique, dealing with contaminated versions of

Table 1
Speech recognition set up parameters

Noise type	SNR (dB)	Noise compensation technique
White	30.18	None
Babble	26.54	VTS
	21.39	
	15.62	

the utterances previously recorded by the speakers. Table 1 summarises the speech recognition set up parameters.

5.2. Scenario and utterance corpora

We have used a dialogue corpus we previously recorded in a fast food restaurant. This corpus contains 523 of dialogues between clients and assistants. We recorded this corpus without the clients being warned that they were recorded. We have created 18 different scenarios that contain the user goals (1–5 product orders) and the user address (phone number, zip code and address data). To create the scenarios 105 sentences were randomly selected in the dialogue corpus representing user goals, user address, confirmations and error indications. In addition to record these sentences we also recorded the sentences the simulator may need to answer the dialogue system correction prompts. It implies that it was necessary to record two additional utterances per food order (corresponding to amount and ingredients), three additional utterances per drink order (corresponding to amount, size and taste) and four additional utterances per address (corresponding to name, building number, floor and letter)—the name refers to the name of the street, avenue, square, etc. Table 2 sets out the number of sentences selected for the eight sentence types, the number of utterances to record corresponding to these sentences, and whether the sentence types are or not explicitly represented as user goals in the scenarios.

For each sentence a phonetic transcription was created. The semantic representations corresponding to these sentences were created automatically from the phonetic transcriptions using the SAPLEN system linguistic analyser. The semantic representations corresponding to the sentences (a)–(e) have been included in the scenarios to represent the user goals.

Nine speakers read sets of the 264 sentences to create the utterance corpus, which contains 1651 utterances. Every sentence was recorded at least six times by at least four different speakers. Four of the speakers spoke standard Spanish, four spoke Spanish from southern Spain, and one speaker was a Japanese female who speaks Spanish. The utterances were recorded under non-noisy lab conditions using a PC computer, using 16

Table 2
Sentence selection to create the utterance corpus

Sentence type	No. of different sentences selected in the dialogue corpus	No. of recorded utterances corresponding to the sentences	Scenario goal
(a) Food order	21	63	Yes
(b) Drink order	15	60	Yes
(c) Phone number	18	18	Yes
(d) Zip code	18	18	Yes
(e) Address	18	90	Yes
(f) Affirmative confirmation	5	5	No
(g) Negative confirmation	5	5	No
(h) Error indication	5	5	No
Total	105	264	

bits/sample at 8 KHz. The speakers were told to record the utterances speaking spontaneously, using hesitation words (as uh) if they wanted. It took us about 2 weeks to prepare the scenario and utterance corpora.

It would be very difficult to record all the possible utterances the user may utter to interact with the dialogue system in a real situation. We recorded only those utterances the simulator may need to achieve the goals defined in the 18 scenarios. Consequently, the corpora are not complete at all, and the evaluation results may not be very real. In order to obtain more refined results much more scenarios and utterances should be included in the corpora. It must be mentioned that the aim of this paper is not showing accurate evaluation results. Instead, the main goal is showing how the ADG technique works and how it could be used for testing dialogue systems.

5.3. Dialogue system evaluation

In order to evaluate the system using the ADG technique, we have focused on word accuracy (WA), sentence understanding (SU) and task completion (TC). The WA is calculated as $\mathrm{WA} = w_t - (w_i + w_s + w_d)/w_t$, where w_t is the total number of words in the utterances used by the simulator to generate its responses, and w_i, w_s and w_d are the number of words inserted, substituted and deleted by the recogniser, respectively. The SU is calculated as the percentage of utterances actually understood by the dialogue system, i.e. $\mathrm{SU} = S_u/S_t$, where S_u is the number of utterances correctly understood by the dialogue system, and S_t is the total number of utterances generated by the simulator. The dialogues between the SAPLEN system and the simulator have been cancelled if the simulator makes more than 30 interactions. We have considered that a generated dialogue has completed the task when the dialogue system understands correctly all the goals in the corresponding scenario. The TC represents the percentage of task-completed dialogues. So $\mathrm{TC} = (D_t - (D_c + D_f))/D_t$, where D_t is the total number of generated dialogues, D_c is the number of cancelled dialogues (because the 30-interaction limit was exceeded), and D_f is the number of dialogues not properly finished because some user goal was not correctly understood by the dialogue system. One hundred dialogues have been generated for each scenario considering all the combinations of the noise conditions and compensation methods specified in Table 1, which accounts for a total of 28,800 automatically generated dialogues.

5.3.1. Word accuracy

Fig. 4 shows the results obtained concerning WA.

The average scores considering all the scenarios for every SNR are shown in Table 3.

The average WA without VTS considering all the scenarios and all the SNR values is 83.07% for the white noise and 81.38% for the babble noise. Using VTS, the average WA considering all the scenarios and all the SNR values is 88.44% for the white noise and 86.72% for the babble noise.

5.3.2. Sentence understanding

The SAPLEN system linguistic analyser uses an implicit recovery strategy that in some occasions, permits to obtain the correct semantic representation in spite of the fact that

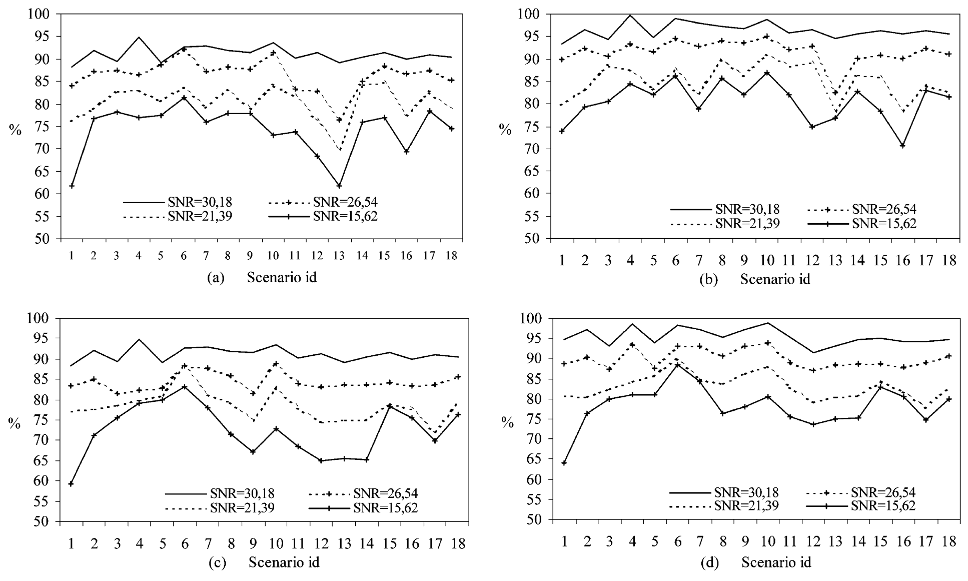

Fig. 4. WA results. (a) White noise without VTS, (b) white noise with VTS, (c) babble noise without VTS, (d) babble noise with VTS.

some words in the recogniser output may be wrong. On the one hand, the strategy recovers meaningless words' recognition errors. As these words do not affect the meaning, their insertions, deletions or substitutions for other meaningless words are easily recovered. In addition, the strategy recovers Spanish gender/number recognition errors, as these lexical mistakes do not affect the conceptual meaning of words. Fig. 5 shows the results obtained concerning SU, including recognition errors that have been implicitly recovered.

The average scores considering all the scenarios for every SNR are shown in Table 4.

The average SU without VTS considering all the scenarios and all the SNR values is 70.1% for the white noise and 68.8% for the babble noise. Using VTS, the average WA considering all the scenarios and all the SNR values is 81.79% for the white noise and 80.61% for the babble noise.

5.3.3. Task completion

Fig. 6 shows the results obtained concerning TC.

Table 3
WA average results

SNR	White noise				Babble noise			
	30.18	26.54	21.39	15.62	30.18	26.54	21.39	15.62
No VTS	90.42	84.39	73.41	72.31	91.13	86.51	80.38	74.26
VTS	96.35	91.67	85.20	80.54	95.45	90.06	83.20	78.19

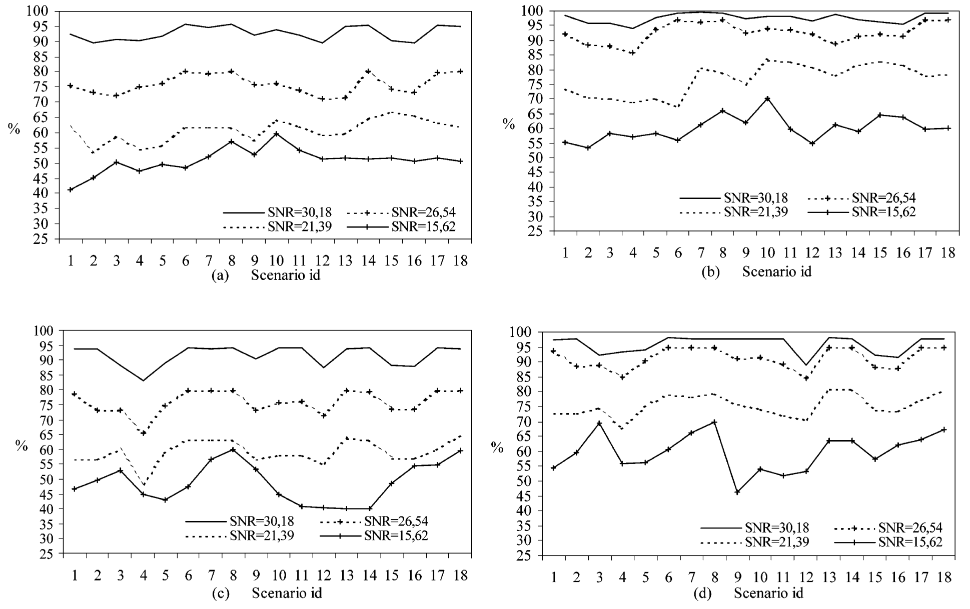

Fig. 5. SU results. (a) White noise without VTS, (b) white noise with VTS, (c) babble noise without VTS, (d) babble noise with VTS.

The TC average scores considering all the scenarios for every SNR are shown in Table 5.

The average TC without VTS considering all the scenarios and all the SNR values is 31.43% for the white noise and 29.64% for the babble noise. Using VTS, the average TC considering all the scenarios and all the SNR values is 57.35% for the white noise and 54.44% for the babble noise.

Observing Table 5 and taking into account the SNR values considered as well as the noise types tested, it is possible to obtain a user-acceptability estimation in noise conditions for the experimental set up in terms of TC. When VTS is not used the system performance cannot be considered acceptable if SNR = 30.18 dB or lower, as TC would be lower than 31%, i.e. less than 1/3 of the generated dialogues would end successfully. If VTS is used the system performance can be considered acceptable if SNR = 26.54 dB or higher, as the least TC in this case would be approximately 90%, i.e. approximately 9/10

Table 4
SU average results

SNR	White noise				Babble noise			
	30.18	26.54	21.39	15.62	30.18	26.54	21.39	15.62
No VTS	92.63	76.02	60.83	50.9	91.51	75.91	59.02	48.77
VTS	97.63	92.65	76.83	60.04	95.92	91.24	75.47	59.78

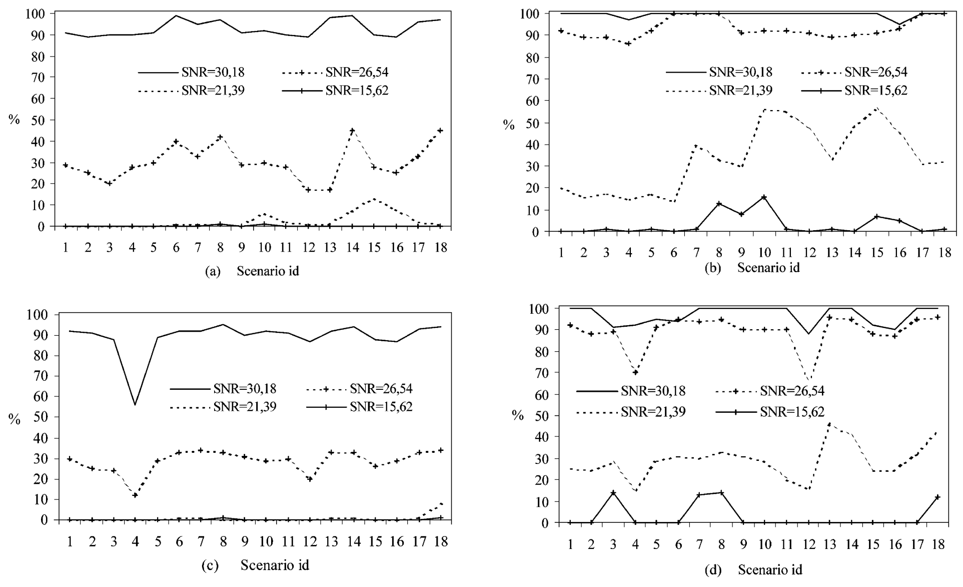

Fig. 6. TC results. (a) White noise without VTS, (b) white noise with VTS, (c) babble noise without VTS, (d) babble noise with VTS.

of the generated dialogues would end successfully. It must be noticed that this estimation may be only valid for the experimental set up and may not the same in the real world.

5.4. Discussion of the results obtained

In this section, we compare the results obtained concerning TC with those reported by Lamel et al. (2000) concerning dialogue success, and explain how the SAPLEN system would work if the interaction simulator-system would be based on either the phonetic transcriptions or the semantic representations.

Lamel et al. (2000) present the evaluation of a dialogue system in terms of dialogue success as a function of word error rate. The dialogue success rates are determined considering whether the dialogue system responses are correct. It could be possible to consider dialogue success in our evaluation, thinking that SAPLEN responses are correct

Table 5
TC average results

SNR	White noise				Babble noise			
	30.18	26.54	21.39	15.62	30.18	26.54	21.39	15.62
No VTS	92.24	30.22	2.44	0.11	89.06	28.78	0.72	0
VTS	99.56	93.17	33.61	3.06	96.78	89.22	28.83	2.94

if the system understand correctly what the simulator says, i.e. if the semantic representations it obtained are correct. Then, it would be possible to map dialogue success to sentence understanding and compare the results presented in this paper concerned with TC with those presented in Lamel et al. (2000) related to dialogue success. Table 5 indicates that under white noise and using VTS, very low TC scores are obtained for SNR = 21.39 dB and SNR = 15.62 dB (33.61 and 3.06%, respectively). Table 3 shows that these scores are obtained for WA = 85.20 and 80.54%, i.e. for word error rates (WER) ≈ 15 and ≈ 20%, respectively. Table 4 sets out that the SU scores obtained for these WER are ≈ 77 and ≈ 60%, respectively. Lamel et al. (2000) report that when WER is ≈ 15% the dialogue success is ≈ 82% and when WER is ≈ 20% the dialogue success is ≈ 78%. Comparing these scores concerning both systems it follows that the performance of the SAPLEN system in terms of dialogue success is ≈ 5% lower if WER ≈ 15 and ≈ 18% lower if WER ≈ 20%.

If the simulator communicates directly with the dialogue system in the semantic representation using frames, i.e. recognition errors are not considered, then TC would be 100%. If the interaction would be based on the phonetic transcriptions, SU would be also 100% and TC would be 100% again. These scores would be obtained because the correct semantic representations (frames) corresponding to the utterances are obtained from the analysis of the phonetic transcriptions, and these frames are included in the scenarios to represent the goals. Consequently, if the interaction is based on the phonetic transcriptions all the semantic representations obtained by the dialogue system are correct and the simulator achieves all the goals. The same happens if the interaction is based on the semantic frames. In this case, all the semantic representations obtained by the dialogue system are correct, as they are those provided by the simulator. The really interesting interaction is when the dialogue system must handle semantic representations built from recognition errors.

5.5. *Using the ADG technique for finding errors in the dialogue strategies employed*

Scheffler and Young (2000) present a user simulator to test a dialogue system designed to carry out bank transfers. This paper indicates the simulations revealed an unacceptably high failure rate for transfer transaction due to a high recognition error rate on cash amounts. This was a flaw in the dialogue strategy that escaped the developers' attention until it was highlighted by the simulation. Similarly, the ADG technique has let us find out a flaw in the dialogue system concerning the confirmation strategy. To find out this problem we have focused on the five scenarios of lowest average TC considering all the SNR values with and without VTS. These scenarios are no. 2, 3, 4, 5 and 12. We have observed that when the SNR is low there are many recognition errors in the confirmation turns. For example, in occasions the dialogue system is asking to confirm a phone number that is correctly recognised and the simulator answers "*yes*" to confirm it. However, in occasions the recogniser output is "*no*" instead of "*yes*", as can be observed in the Example 6, causing that the dialogue system takes more turns to obtain the phone number.

Speaker	**Utterance**	**Simulator action**
SYSTEM:	Please, say your phone number	-
SIMULATOR:	9 5 8 1 2 3 1 4 7	Use a scenario goal
SYSTEM:	Did you say 9 5 8 1 2 3 1 4 7?	-
SIMULATOR:	Yes	Confirmation
SYSTEM:	Ok, please, repeat your phone number	-

This problem can make the dialogue system exceed the interaction limit, therefore some generated dialogues fail to achieve the goals, resulting in a low TC. In other occasions, the simulator answers "*no*" to indicate that the obtained phone number is wrong, but the recogniser output is "*yes*". This error makes that the dialogue system wrongly believes the phone number is confirmed, as can be seen in the Example 7.

Speaker	**Utterance**	**Simulator action**
SYSTEM:	Please, say your phone number	-
SIMULATOR:	9 5 8 1 2 5 1 4 9	Use a scenario goal
SYSTEM:	Did you say 9 5 8 6 9 3 8 5 7?	-
SIMULATOR:	No	Correction
SYSTEM:	Ok, would you like to order for anything else?	-

These problems indicate that it is necessary to set up a new strategy to handle confirmations. A possible solution would be to ask again the user as could be observed in the Example 8.

Speaker	**Utterance**	**Simulator action**
SYSTEM:	Please, say your phone number	-
SIMULATOR:	9 5 8 1 2 3 1 4 7	Use a scenario goal
SYSTEM:	Did you say 9 5 8 1 2 3 1 4 7?	-
SIMULATOR:	Yes	Confirmation
SYSTEM:	Did you say yes?	-
SIMULATOR:	Yes	Redundant confirmation
SYSTEM:	Ok, would you like to order for anything else?	-

This new strategy may help to avoid data to be wrongly confirmed. However, it would lengthen the dialogues dramatically then it would be preferable to explore other strategies. We have observed also that in occasions dialogues exceed the interaction limit because recognition errors force the dialogue system to prompt for the same data repeatedly. Most of the times, this error occurs during the recognition of the user data (for example, a digit in the phone number). In real dialogues this problem would annoy the user, then it would be preferable to use other strategy. For example, in case of very poor recognition, it would be preferable changing the input mode (i.e. to DTMF) or calling a human operator.

6. Limitations of the ADG technique

A limitation is that in real conversations users may behave almost unexpectedly and may not always answer the system prompts in order to provide the data the system asks for. However, the simulator always provides the data when the dialogue system prompts for it.

Also, the simulator has no doubts during the interaction and never gets either confused or nervous because of the use of a machine, which is not always the case of a real user. Additionally, real users may modify their interaction when they face to very poor recognition. For example, they may reduce the number of things they say per utterance, spell names or speak slower/louder.

Other limitation is that the simulator always provides all the data the system needs for understanding a goal completely. For example, it may say, "*I want one ham sandwich please*" instead of just saying "*one sandwich please*". This simulator behaviour was decided for simplification purposes but, of course, real users are not necessarily so prolific when they interact to a machine. Also, at the moment the simulator can only order for one-product per turn, but a real user may order for all the products he/she wants in just one turn. The reason for this restriction is that we initially thought of concatenating recorded utterances concerning one-product orders to artificially obtain utterances concerning several-product orders. However, this procedure for utterance concatenation is not yet set up. The way the simulator generates confirmations constitutes other limitation. The procedure set up by now is based only on "*yes*"/"*no*" answers. For example, in the (3) turn of the Example 5, the simulator answered "no" but a real user would have said "*no, I said 9 5 8 2 7 5 3 6 0*".

Taking into account these limitations of the ADG technique it is clear that the ability of the simulator to really mimic the behaviour of a real user is, by now, very limited. However, it would not be very difficult to refine the procedures set up to incorporate more real-word phenomena into the model, as it is described in Section 8.

7. Conclusions

This paper presents our first contribution to the challenging problem of simulating a user interacting with a dialogue system. In despite of the limitations, the paper shows that the ADG technique can be used to test spoken dialogue systems under different lab-generated conditions. A dialogue system has been evaluated considering white and babble noise as well as a VTS noise compensation technique. The results prove the system works worse under babble noise. The VTS technique has been effective to deal with noisy utterances and has led to better experimental results, especially under white noise. The results presented in Table 5 indicate that when VTS is not used the system can be considered acceptable if SNR is at least 30.18 dB, as TC is close to 90%. When VTS is used the system can be considered acceptable when SRN is at least 26.54 dB. From these results, it is possible to conclude that if TC close to 90% is considered acceptable for users, then SU should be at least 91.24% and WA should be at least 90.06%. These results are valid for the task studied and the interaction setting, which may not reproduce exactly a real dialogue between a user and the dialogue system.

The ADG technique permitted to detect problems in the recognition and dialogue management, by focusing on the generated dialogues with low TC. There are two main reasons for low TC in the experiments. On the one hand, errors in the recognition of confirmations generate cancelled dialogues, wrong confirmations of data, and continuous repetitions of prompts for the same data. On the other hand, other recognition errors can provoke inappropriate focus shifting, making some scenario goals to be incorrectly understood by the system.

8. Future work

The are several issues that need to be addressed in order to make the ADG technique able to generate dialogues more similar to real ones. On the one hand, the technique relies on a set of scenarios that represent user goals. It would be possible to include new types of goals in the scenarios as an attempt to simulate more aspects of a real user-system interaction, as for example, user change of mind, negotiation, digression, etc.

In order to set up different ways for the simulator to utter a goal it would be possible to consider two types of slot in the goals, let us call them 'main' slots and 'secondary' slots. The main slots would represent the minimum information a user may utter to achieve a goal and the secondary slots would represent the information a user may not utter in a first attempt to achieve a goal but that should utter later. The procedure for utterance selection could be enhanced to let the simulator decide whether to use either all the slots concerning a goal, just the main slots or a combination of main and secondary slots. This would be a way of modelling that real users do not always utter all the data the system needs to understand a goal, as well as a way of modelling that users usually shorten their utterances when they face to poor recognition. Additionally, utterances recorded slower or louder could be used as a way of modelling that users may speak differently when communication problems appear. Currently the simulator only generates "*Yes*" or "*No*" responses for the confirmations generated by the dialogue system. This procedure could be enhanced to incorporate a mixed initiative that lets the simulator generate responses as "*No, I said…*", commonly uttered by real users. These changes to enhance the ADG technique would require recording new utterances corresponding to the different ways the simulator would have to express a goal or a confirmation. It would also require preparing the phonetic transcriptions and semantic representations corresponding to these new utterances.

A procedure for concatenating utterances at the voice signal and the phonetic transcription levels could be set up for concatenating one-product orders to artificially obtain several-product orders. For example, the utterances: "*one ham sandwich*", "*one big beer*", "*and*" and "*one vegetal and turkey salad*" recorded independently could be concatenated to obtain the utterance "one ham sandwich, one big beer and one vegetal and turkey salad". Although it would not be the same for the recogniser to analyse the concatenated utterances and real ones, it would be an easy method to obtain different product orders without the need to explicitly record each one. To represent a several-product order goal in a scenario it would be possible to use the semantic representation obtained by the linguistic analyser from the analysis of the concatenated phonetic transcription. The procedure for goal selection would decide whether a goal refers to a single-product order or a several-product order by examining the slots. The first case would be handled as it is processed now and the second one would require using the new procedure for the utterance concatenation.

References

Albasano, D., Baggia, P., Danieli, M., Gemello, R., Gerbino, E., Rullent, C., 1997. DIALOGOS: A Robust System for Human–Machine Spoken Dialogue on the Telephone, Proceedings of International Conference on Acoustics, Speech and Signal Processing, pp. 1147–1150.

Allen, J., 1995. Natural Language Understanding, The Benjamin/Cummings Publishing Company Inc, Menlo Park.

Ammicht, E., Gorin, A., Alonso, T., 1999. Knowledge Collection for Natural Language Spoken Dialog Systems, Proceedings of Eurospeech, pp. 1375–1378.

Bernsen, N.O., Dybkjaer, L., Heid, U., 1999. Current Practice in the Development and Evaluation of Spoken Language Dialogue Systems, Proceedings of Eurospeech, pp. 1147–1150.

Bouwman, G., Hulstijn, J., 1998. Dialogue Strategies Redesign with Reliability Measures, Proceedings of First International Conference on Language Resources and Evaluation, pp. 191–198.

Brieussel-Pousse, L., Perennou, G., 1999. Language Model Level vs. Lexical level for Modeling Pronunciation Variation in a French CSR Speech, Proceedings of Eurospeech, pp. 1771–1774.

Casacuberta, F., García, R., Llisterri, J., Nadeu, C., Pardo, J.M., Rubio, A., September 1991. Development of Spanish Corpora for Speech Research (ALBAYZIN). Proceedings of the Workshop on International Cooperation and Standardization of Speech Databases and Speech I/O Assessment Methods. Chiavari, Italy, pp. 26–28.

Eckert, W., Levin, E., Pieraccini, R., 1997. User modelling for spoken dialogue system evaluation, Proceedings of IEEE ASR Workshop.

Eckert, W., Levin, E., Pieraccini, R., 1998. Automatic evaluation of spoken dialogue systems. Technical Report, TR98.9.1, AT and T Labs Research.

Edgington, M., Attwater, D., Durston, P., 1999. OASIS—A Framework for Spoken Language Call Steering, Proceedings of Eurospeech, pp. 923–926.

Ehrlich, U., 1999. Task Hierarchies Representing Sub-Dialogs in Speech Dialog Systems, Proceedings of Eurospeech, pp. 1387–1390.

Ehsani, F., Bernstein, J., Najmi, A., 2000. An interactive dialog system for learning Japanese. IEEE Transactions on Speech and Audio Processing 8, 167–177.

Fedrico, M., 2000. A system for the retrieval of Italian broadcast new. IEEE Transactions on Speech and Audio Proccessing 8, 37–47.

Gibbon, D., Mertins, I., Moore, R.K., 2000. Handbook of Multimodal and Spoken Dialogue Systems, Kluwer Academic Publishers, Dordrecht.

Hain, T., Woodland, P.C., Niesler, T.R., Whittaker, E.W.D., 1999. The 1998 HTK System for Transcription of Conversational Telephone Speech, Proceedings of International Conference on Acoustics, Speech and Signal Processing.

Krahmer, E., Swerts, M., Theune, M., Weegels, M., 1999. Problem Spotting in Human–Machine Interaction, Proceedings of Eurospeech, pp. 1423–1426.

Lamel, L., Rosset, S., Gauvain, J.L., Bennacef, S., Garnier-Rizet, M., Prouts, B., 2000. The LIMSI arise system. Speech Communication 31, 339–353.

Lee, C., Carpenter, B., Chou, W., Chu-Carroll, J., Reichl, W., Saad, A., Zhou, Q., 2000. On natural language call routing. Speech Communication 31, 309–320.

Levin, E., Pieraccini, R., Eckert, W., 2000. A stochastic model of human–machine interaction for learning dialog strategies. IEEE Transactions on Speech and Audio Processing 8, 11–23.

Lin, B., Wang, H., Lee, L., 1999. Consistent Dialogue Across Concurrent Topics Based on an Expert System Model, Proceedings of Eurospeech, pp. 1427–1430.

López-Cózar, R., Rubio, A.J., García, P., Segura, J.C., 1999. A New Word-Confidence Threshold

Technique to Enhance the Performance of Spoken Dialogue Systems, Proceedings of Eurospeech, pp. 1395–1398.

López-Cózar, R., Rubio, A.J., Díaz Verdejo, J.E., De la Torre, A., 2000a. Evaluation of a Dialogue System Based on a Generic Model that Combines Robust Speech Understanding and Mixed-Initiative Control, Proceedings of First International Conference on Language Resources and Evaluation, pp. 743–748.

López-Cózar, R., Rubio, A.J., Benítez, M.C., Milone, D.H., 2000b. Restricciones de Funcionamiento en Tiempo Real de un Sistema Automático de Diálogo (Real-Time Performance of a Spoken Dialogue System). Spanish Society for Natural Language Processing Journal 26, 169–174.

McAllaster, D., Gillick, L., 1999. Studies in Acoustic Training and Language Modeling Using Simulated Speech Data, Proceedings of Eurospeech, pp. 1787–1790.

McTear, M.F., 1999. Software to Support Research and Development of Spoken Dialogue Systems, Proceedings of Eurospeech, pp. 339–342.

McTear, M.F., Allen, S., Clatwrthy, L., Ellison, N., Lavelle, C., McCaffery, H., 2000. Integrating Flexibility into a Structured Dialogue Model: Some Design Considerations, Proceedings of International Conference on Speech and Language Processing, pp. 943–946.

Meng, H.M., Wai, C., Pieraccini, R., 2000. The use of belief networks for mixed-initiative dialog modeling, Proceedings of International Conference on Speech and Language Processing, pp. 106–109.

Moreno, P.J., 1996. Speech recognition in noisy environments. PhD Thesis, CMU.

Müller, C., Schröder, K., 1999. Standardised Speech Interfaces—Key for Objective Evaluation of Recognition Accuracy, Proceedings of Eurospeech, pp. 931–934.

Nasr, A., Esteve, Y., Béchet, F., Spriet, T., de Mori, R., 1999. A Language Model Combining N-grams and Stochastic Finite State Automata, Proceedings of Eurospeech, pp. 2175–2178.

Niesler, T., Woodland, P., 1999. Variable-length category n-gram language models. Computer, Speech and Language 21, 1–26.

Niimi, Y., Oku, T., Nishimoto, T., Araki, M., 2000. A task-independent dialogue controller based on the extended frame-driven method, Proceedings of International Conference on Speech and Language Processing, pp. 114–117.

Pargellis, A., Kuo, J., Lee, C.H., 1999. Automatic Dialogue Generator Creates User Defined Applications, Proceedings of Eurospeech, pp. 1175–1178.

Pfau, T., Faltlhauser, R., Ruske, G., 1999. Speaker Normalization and Pronunciation Variant Modeling: Helpful Methods for Improving Recognition of Fast Speech, Proceedings of Eurospeech, pp. 299–302.

Polifroni, J., Seneff, S., 2000. Galaxy-II as an Architecture for Spoken Dialogue Evaluation, Proceedings of Second International Conference on Language Resources and Evaluation, pp. 725–730.

Polifroni, J., Seneff, S., Glass, J., Hazen, T.J., 1998. Evaluation Methodology for a Telephone-Based Conversational System, Proceedings of First International Conference on Language Resources and Evaluation, pp. 43–49.

Scheffler, K., Young, S., 2000. Probabilistic Simulation of Human–Machine Dialogues, Proceedings of International Conference on Acoustics, Speech and Signal Processing, pp. 1217–1220.

Siu, M., Ostendorf, M., 2000. Variable N-Grams and Extensions for Conversational Speech Language Modeling. IEEE Transactions on Speech and Audio Processing, 63–75.

Smadli, K., Brun, A., Zitouni, I., Haton, J.P., 1999. Automatic and Manual Clustering for Large Vocabulary Speech Recognition: A Comparative Study, Proceedings of Eurospeech, pp. 1795–1798.

Souvignier, B., Kellner, A., Rueber, B., Schramm, H., Seide, F., 2000. The thoughtful elephant:

strategies for spoken dialog systems. IEEE Transactions on Speech and Audio Processing 8, 51–62.

Stern, R.M., Raj, B., Moreno, P.J., 1997. Compensation for environmental degradation in automatic speech recognition. ESCA-NATO Workshop on Robust Speech Recognition for Unknown Communication Channels. Pont-a-Musson, France.

Woodland, P.C., Hain, T., Moore, G.L., Niesler, T.R., Provey, D., Tuerk, A., Whittaker, E.W.D., 1999. The 1998 HTK Broadcast News Transcription System: Development and Results. DARPA Broadcast News Workshop.

Zue, V., Seneff, D., Polifroni, J., Phillips, M., Pao, C., Goodine, D., Goddeau, D., Glass, J., 1994. PEGASUS: a spoken dialogue Interface for on-line air travel planning. Speech Communication 15, 331–340.

Zue, V., Seneff, S., Glass, J.R., Polifroni, J., Pao, C., Hazen, T.J., Hetherington, L., 2000. JUPITER: a telephone-based conversational interface for weather information. IEEE Transactions on Speech and Audio Processing vol. 8, no. 1, 85–95.