ISSN 2007-9737

Arabic Dialogue System for Hotel Reservation based on Natural Language Processing Techniques

Asma Moubaiddin[1], Ola Shalbak[2], Bassam Hammo[2], and Nadim Obeid[2]

[1] Department of Linguistics, The University of Jordan, Amman, Jordan

[2] KASIT, CIS Department, The University of Jordan, Amman, Jordan

{a.mobaiddin, o.shalbak, b.hammo, nadim}@ju.edu.jo

Abstract. In this paper, we present an Arabic dialogue system (also referred to as a conversational agent) intended to interact with hotel customers and generate responses about reserving a hotel room and other services. The system uses text-based natural language dialogue to navigate customers to the desired answers. We describe the two main modules used in our system: the parser and the dialogue manager. The parser is based on the Government and Binding theory. Customers can inquire about room availability, hotel services and negotiate a desired reservation. We report an experiment with 500 volunteers unfamiliar with the system in a real environment. The users were asked to interact with the system and then to judge the dialogues as "very bad," "bad," "neutral," "good," or "very good." We found that 66.92% of the dialogues were judged to be "very good" and 92.3% were judged to be "good" or "very good". These results confirm the viability of using an Arabic dialogue system to tackle the problem of interactive Arabic dialogues. Finally, we discuss future directions for enhancing our dialogue system with more sophisticated and intuitive interaction.

Keywords. Dialogue system, conversational agent, computational linguistics, Arabic parser, government and binding theory.

1 Introduction

A dialogue system can be viewed as an advanced application of natural language processing technology. In literature, many solutions to dialogue systems have been introduced. They have employed text, speech, graphics, gestures, and other modes for communication on both the input and output channels [17]. The objective of developing a dialogue system is to provide a human-centric interface between a user (e.g. human or computational agent) and a computer (system) to access and manage information [23]. Many (spoken or written) dialogue systems have been developed to support applications from different domains. Examples may include flight information services, theatre ticket booking, weather information, travel planning, and smart home [1, 5, 9, 16, 19, 21, 22, 26, 27, 29, 31, 33].

A text-based dialogue system receives natural language inputs from a user and gives an appropriate response/action. It should be able to engage in a well structured dialogue where it can take control in order to confirm information given by the user, clarify the situation, or constrain the user responses if misunderstandings arise. It has to respond properly and offer appropriate information as required. This requires the ability to interpret user responses according to the dialogue context in order to correctly understand the user intention.

In this paper, we present an Arabic dialogue system (ADS) intended to interact with hotel customers and generate responses about reserving a hotel room and other services. The system uses text-based natural language dialogue to lead customers to the desired answers. The system consists of a parser for the Arabic language and a dialogue manager (DM). It

receives textual inputs from the user, interprets the message, and responds with the required action and/or information. Our system combines natural language understanding and flexible dialogue control to enable natural conversational interaction by users to access information about hotel services and room availability, and to make reservations. Specifically, users can use natural dialogue to negotiate a desired reservation. Users can inquire about room availability and pricing as well as obtain information about appropriate services provided by the hotel. We have decided to employ the Government and Binding (GB) theory to implement a GB-based parser [3, 13, 14, 25]. Before we discuss the parser, we will give a brief presentation of the Arabic language and the GB theory.

The rest of the paper is organized as follows. In Section 2 we give some background on the Arabic language and the Government and Binding (GB) theory. Section 3 presents some related works. Section 4 is about the system's architecture, while Section 5 goes over the implementation part. Section 6 sheds light on ADS components as compared to some other systems. Section 7 shows the experiments and the evaluation of ADS, and finally, Section 8 presents the conclusions and future work directions.

2 Some Background on Arabic and the Government and Binding Theory

Arabic is a Semitic language that has a rich morphology and a flexible word order. In this paper we are concerned with Modern Standard Arabic (MSA), which is used in modern writing and is understood by Arabic speakers. The Arabic grammar distinguishes between two types of sentences, verbal and nominal. Nominal sentences have two parts: a subject (*mobtada?* مبتدأ) and a predicate (*khabar* خبر).

When a nominal sentence is about being, i.e. if the verb of the sentence is 'to be' in English, this verb is not given in Arabic. The Arabic morphology is based on roots and patterns through which words are derived. An Arabic word may be composed of a stem consisting of a base root and a pattern which defines its semantic and syntactical role. Moreover, affixes and clitics are often attached to words.

Affixes include inflectional markers of tense, gender, and number. Clitics include prepositions, conjunctions, determiners, and possessive pronouns. Here are some of the characteristics of the Arabic language:

1. It has a relatively free word order. It is not uncommon to find VSO, SVO, and VOS word orders within an Arabic text as in the following examples:
   - **قرأَ المعلمُ الدرسَ** (***the lesson*** / ***the teacher*** / ***read***)
     *qara?-a al-mu*ʕ*alim-u a-dars-a: VSO*
   - **المعلمُ قرأَ الدرسَ** (***the lesson*** / ***read*** / ***the teacher***)
     *al-mu*ʕ*alim-u qara?-a a-dars-a: SVO*
   - **قرأَ الدرسَ المعلمُ** (***the teacher*** / ***the lesson*** / ***read***)
     *qara?-a a-dars-a al-mu*ʕ*alim-u: VOS*

   All of the sentences above are grammatical and have the same English meaning of "the teacher reads the lesson".
2. Arabic is a clitic or clitic-directed language. Clitics are morphemes that have syntactic characteristics of a word but are morphologically bound to other words (e.g., coordinating conjunctions, the definite article, many prepositions and particles, and a class of pronouns that are attached either to the start or end of words) as in كتبنا *katabna* "we wrote*"* which is made up of the verb كتب: *katab* and the clitic نا: *na* which acts as the subject for the verb كتب *katab* "wrote".
3. The omission of diacritics (syntactic marks) in most written Arabic texts.
4. Arabic is a pro-drop language. The subject can be omitted leaving any syntactic parser with the challenge to decide whether or not there is an omitted pronoun in the subject position.
5. Homographs of words with/without the same pronunciation are often produced. They have different meanings and usually different POS. For example, the word ذهب can be interpreted as *thahab-a* (verb) meaning "he went" or *thahab* (noun) meaning "gold".

Government and Binding (GB) Theory is an approach to Universal Grammar which includes rules and principles that apply to all languages [3]. However, while certain grammatical principles and

rules are universal, there is a lot of variation between different languages such as different ordering for subject (S), verb (V), and object (O). It is agreed that every language has a basic word order, and all other word orders result from the movement of sentence constituents and this movement is restricted by some rules and principles [6].

Words are organized hierarchically into bigger units called phrases. Phrase constituents include:

1. IP - Inflectional Phrase: a phrase headed by I. I/INFL stands for Inflection, and it consists of tense, number, and gender agreement (AGR) elements.
2. CP - Complementizer Phrase: a phrase headed by a complementizer (C). C takes an IP (INFL Phrase) as its complement and heads the maximal projection CP.
3. NP - Noun Phrase: a phrase headed by a noun (N).
4. VP - Verb Phrase: a phrase headed by a verb (V).
5. AP - Adjectival Phrase: a phrase headed by an adjective (A).
6. PP - Prepositional Phrase: a phrase headed by a preposition (P).

The main principles of GB Theory include:

1. Government which is concerned with the syntactic relations in a sentence and has its main application in case assignment.
2. Theta Theory which is concerned with describing the thematic relations between arguments and predicates.
3. Predicates and arguments: the arguments are the participants minimally involved in the activity or state expressed by the predicate.
4. Case Theory which is concerned with the assignment of abstract case: nominative, accusative, and genitive, to words, based on their position in a sentence.
5. X-Bar Theory which is concerned with phrase formation. It states that all phrases are headed by a lexical head (noun, verb, adjective, or preposition).
6. Complements combine with X to form X' projections, adjuncts combine with X' to form X'. A specifier combines with the topmost X' to form the maximal projection X''/XP.
7. D-structure and S-structure: all sentences are represented in terms of both forms, the D-structure and the S-structure. The D-structure encodes the predicate-argument relations and the thematic properties of a sentence, and it is built upon the *basic word order*. The S-structure accounts for the surface ordering of the sentence constituents.
8. NP-Movement: GB Theory assumes that the different word orders arise from the movement of sentence constituents. Hence, a basic word order is assumed, and all other word orders are derived.

3 Related Work

Some of the early dialogue systems such as ARISE [20] were limited to structured tasks and goals, and to regulated strong-typed interactions [24]. Such systems have been deployed in various practical cases mainly due to their simplicity. However, designing appropriate rules in advance is difficult and the system may lack flexibility. For example, if a user provides more information than required by the system, the system may fail to manage the dialogue flow. Furthermore, domain portability may be poor in such systems, so the designers may have to restart the whole design and development process for every new domain.

To deal with these limitations, many proposals explore generic dialogue models which are based on agenda or task models represented using Information States (IS) [4, 21]. For instance, [4] makes an attempt to separate between the domain-dependent and the domain-independent aspects of the dialogue control logic. The former aspects can be captured by the dialogue task specification. A reusable domain-independent dialogue engine manages the conversation by executing a given dialogue task specification. The use of IS by DM allows us to capture the system's functionalities as a set of actions performed by the application system. The input can be defined in terms of a set of dialogue moves/objects, and the output captures the changes imposed on the dialogue objects and the creation of new ones expressing the performed actions [15]. It is the task of the DM to decide the most appropriate

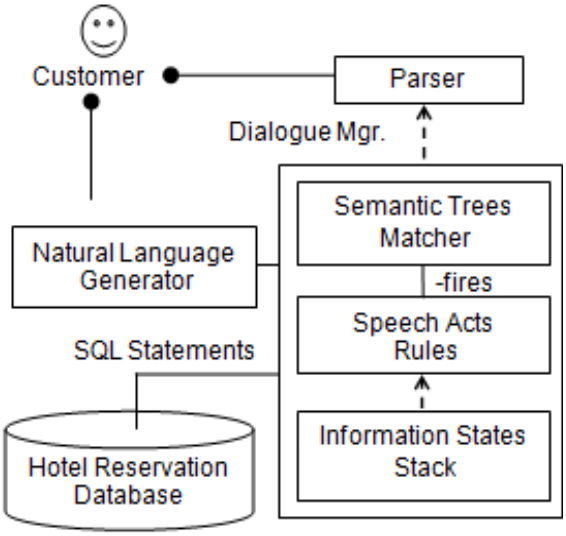

Fig. 1. The ADS architecture

next move to be made by the system by maintaining an appropriate IS.

To the best of our knowledge, there is no published work on Arabic dialogue systems. Shala *et al.* [28] presented an automated method for the task of speech act classification for Arabic discourse. The idea is based on assigning a category from a set of predefined speech act categories to the initial words (or their part-of-speech) in a sentence to indicate speaker's intention. They used named entities and a machine learning algorithm to automatically derive the parameters of the models used to implement their approach based on a corpus of 408 Arabic sentences.

Therefore, in the absence of a gold-standard test bed for Arabic dialogue systems, it was difficult to compare our system to an existing one. Our attempt to test and evaluate ADS falls under the black box method (cf. Section 7.1).

4 Overview of the System Architecture

The Arabic dialogue system for hotel reservation (ADS) is depicted in Fig. 1. It consists of a GB-based parser and a dialogue manager (DM). These components work together. The parser (discussed in Section 4.1) produces a parse tree of the user input. The dialogue manager employs the semantic representation matcher to map a parse tree to a meaning representation in order to interpret the dialogue act and understand the user intention. It fills in the missing arguments through an interaction mode with the user in the predicate which represents the logic form of an SQL query to be submitted to the database. The system generates its responses in Arabic.

4.1 The GB-Based Parser

Unlike rule-based grammars that use a large number of rules to describe patterns in a language, the GB Theory [12] explains these patterns in terms of more fundamental and universal principles. A key issue in building a GB-based parser is how to procedurally interpret the principles expressed as grammar rules. Since the GB principles are constraints over syntactic structures, one way to implement the principles is as follows:

1. Generate candidate structures of a given sentence that satisfy the X-bar Theory and sub-categorization frames of the words in the sentence.
2. Filter out structures that violate any one of the principles.
3. The remaining structures are accepted as the parse trees of the sentence.

The parser takes as input an Arabic sentence (with or without diacritics) represented as a sequence of tags and returns as output the valid syntactic structure(s) of the sentence [13, 14]. In other words, the parser checks the grammaticality of tagged sentences using the GB-based grammar rules and consequently constructs their syntactic structure if they are grammatically valid. We adopt a top-down recursive approach. The implementation is restricted to the following rules:

1. Basic sentence structures for SVO, VOS, and VSO sentences where the subject is an NP and the Object is an NP.
2. Sentences followed by a PP adjunct.
3. Nominal sentences made up from NP(s), or NP followed by PP, and nominal sentences preceded by Inna (or sisters' particles).

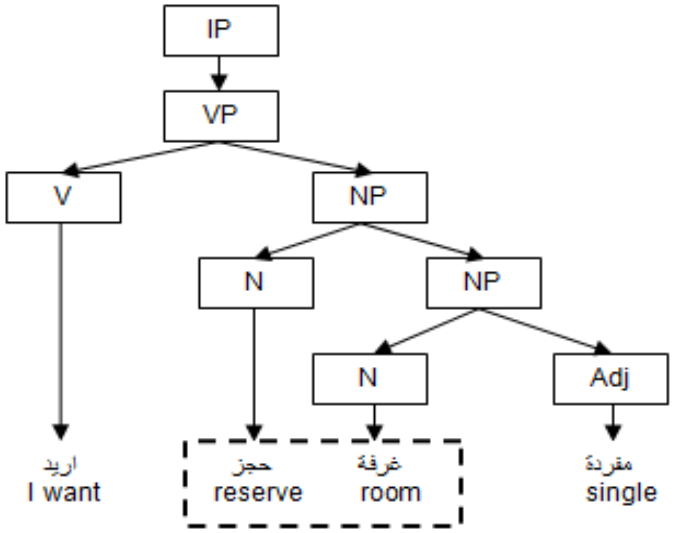

Fig. 2. Parse tree nodes for حجز, *hajez* "reserve" & غرفة, *ghorfah* "room"

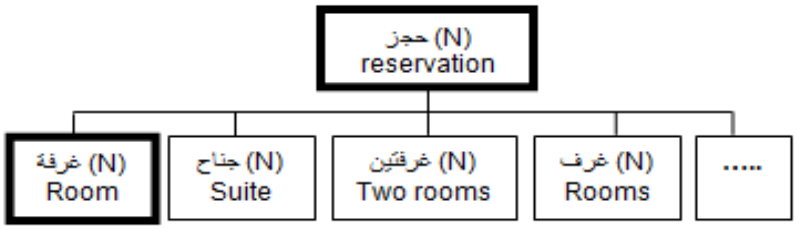

Fig. 3. A semantic tree with a root node that matches the nodes in the parse tree

4. Question Sentences staring with a question word followed by a VSO order sentence.

4.2 Dialogue Manager, Information State, Semantic Trees

The Dialogue Manager (DM) is the core of our dialogue system. It coordinates the activity of all components, controls the dialogue flow, and communicates with the database. Its major tasks include:

– Accepting and understanding the user message(s) to find out the user intention which can be captured using SRTs.
– Querying the hotel database based on the current input and the dialogue context.
– Asking a further question in order to formulate an appropriate query.
– Requesting to confirm unclear information and to rephrase if the user input is inappropriate.
– Predicting the next system move at the concept level to output the system's utterance in Arabic. The system's response(s) has/have to reflect the dialogue context by maintaining the dialogue Information States (IS).
– The IS keeps track of what the participants (e.g., user and system) exchange, i.e., how the dialogue starts, what each participant has at a certain point during the dialogue, and what the expected actions within upcoming moves are. Missing facts that are required to reach the final goal has to be obtained from

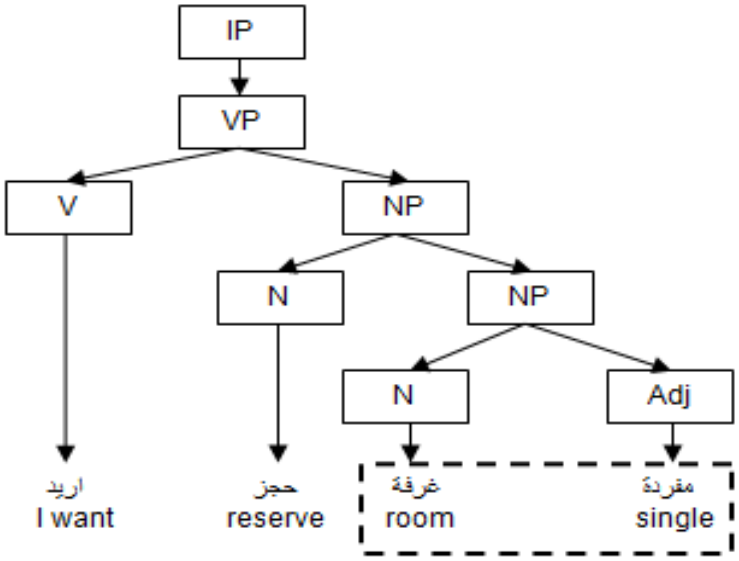

Fig. 4. Next nodes traversed in the parse tree

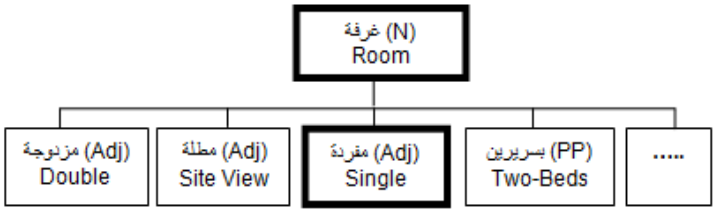

Fig. 5. A semantic tree with غرفة *ghorfah* "room" as the root with a leaf node matching the parse tree

the user. Planning of such an approach is performed by breaking the main task into a set of sub-tasks and fulfilling the goals of these sub-tasks to reach the higher goal.

To interpret the user's message and understand her/his intention, there is a need for semantic categories which are specific to the domain of the dialogue system being developed. We have developed the notion of Semantic Representation Trees (SRTs) which hold the semantic representation of concepts related to the system domain. We have developed SRTs after conducting a survey which aimed at finding out most frequent queries and speech acts/utterances exchanged during dialogues between clients and receptionists in 15 hotels in Jordan.

SRTs are shallow in nature as they hold only a main category and the subcategories to which it is connected. They have the main category in a non-terminal node while terminal nodes hold the user's actual expression [7]. Having the user utterance split into tags (organized as a tree), tree branches can be matched against tree branches of the system's SRTs, in a left-to-right top-down approach (similar to the compilation process of programming languages). An example of an SRT is shown in Fig. 3.

Mapping the parse trees to SRTs can be used to fill in the missing arguments in order to build an SQL query. That is, by traversing the parse tree, mapping each node to the respective terminal node in an SRT and matching the directly

Table 1. ADS stack activities

F(x)	Action
push:	reserve_room (R, True)
push:	**request** ("ما نوع الغرفة التي تريد حجزها", system, RT) *ma naw‹ alghorfah alti toreed hajzaha?* What type of room do you want to reserve?
	inform (single, user, RT)
pop:	RT
push:	**request** ("كم عدد الغرف التي تريد حجزها", system, RN) *km ‹adad alghoraf alti toreed hajzaha?* How many rooms do you want to reserve?
	inform (1, user, RN)
pop:	RN
push:	**request** ("متى تريد الحجز", system, DD) *mta toreed alhajez?* When do you need the reservation?
	inform ("15/12/2014", user, DD)
pop:	DD
push:	**request** ("كم يوما سيكون الحجز", system, RD) *km yawm-n sa-yakoon alhajez?* How many days is the reservation for?
	inform (3, user, RD)
pop:	RD
push:	**request** ("هل لديك استفسارات أخرى", system, EXTRA) *hal ladaik-a estifsarat ukhra?* Do you have any other questions?
	request ("اريد غرفة تطل على البحر", user, Special) *oreed-u ghorfah totel-u ‹ala albhr.* I want a sea view room.
pop:	Special
push:	**request** ("هل لديك استفسارات أخرى", system, EXTRA) *hal ladaik-a estifsarat ukhra?* Do you have any other questions?
	False
pop:	EXTRA
push:	**greet** ("شكرا لاستخدامك خدمات فندقنا", system, false) *shukran le-estekhdamik-a khadamat-i fondokina.* Thank you for choosing our hotel services.
pop:	*R*

R is the room number(s) that was/were reserved when executing Q.

connected branch of the parse tree node to the non-terminal nodes of that of SRT, we can fill in the missing arguments of the predicate which is syntactically captured by the parse tree.

Table 2. Speech acts employed by ADS

Act	Example
Request:	used by either the user or the system to ask for information as in **request** ("متى تريد الحجز", system, DD) *mata toreed-u alhajez?* When do you want to make a reservation?
Inform:	used by the system or the user to provide pieces of needed information or an answer for a given question (e.g., a request), such as **inform** *(date_of_reservation,* user, DD*)* where *X* holds the answer and may take any of the following forms: 15/12/2014 الخميس القادم, *alkhamees alkadem* "Next Thursday" بداية الشهر القادم, *bedayat alshahr alkadem* "The beginning of the next month"
Confirm:	used to make sure that the system (or the user) has got the correct idea or intention. For example, assume that the system's question is متى تريد الحجز؟, *mata toreed-u alhajez?* When do you want to make a reservation? And the user's answer is الخميس القادم, *alkhamees alkadem* Next Thursday Then the system checks the date and triggers the confirm act: **confirm** (15/12/2014 اذا انت تريد الحجز يوم) *ethn anta toreed-u alhajez yawm 15/12/2014?* So you want a reservation on 15/12/2014, don't you?
Greetings:	used as start or end markers of a dialogue. The system may end a dialogue with ***greet*** ("شكرا لاستخدامك خدمات فندقنا", system, false) *shukran le-estekhdamak-a khadamat fondokana* Thank you for choosing our hotel services.

Now consider Figures 2-5 for detailed explanations. Fig. 2 shows the parse tree for the sentence اريد حجز غرفة مفردة, *oreed-u hajez ghorfah mofradah* "I want to reserve a single room", entered by the user requesting reservation غرفة مفردة, *ghorfah mofradeh* "a single room". In Fig. 3, the encircled sub-tree is included in the parse tree of Fig. 2.

Starting at the noun حجز, *hajez* "reservation", there is an SRT which has the noun حجز, *hajez* "reservation" as its root (cf. Fig. 3), it is called SRT-حجز (SRT-*hajez*). The next element of the parse tree is the noun غرفة, *ghorfah* "room", which matches the left-most leaf node of the SRT-حجز. Now, we look for an SRT with root غرفة, root-غرفة (root-*ghorfah*). The third leaf node in root-غرفة is مفردة, *mofradah* "single", as shown in Fig. 4, and there's an SRT with this word as its root. This completes the parse tree traversal process and interpretation (cf. Fig. 5.)

In addition, nodes in the semantic trees have propositions connected to them. For instance, when the user asks أريد حجز غرفة, *oreed-u hajez ghorfah* "I want to reserve a room", the noun: حجز, hajez "reserve", is connected with the proposition S which is as follows:

S = "*reserve_room (R, True)*".

Here, S is pushed into the IS stack as the ultimate goal of the current dialogue which triggers the initiation of an SQL query Q stating the need to search the hotel database for an available room. For Q to be executed, some variables need to be instantiated. DM pushes a set of speech acts

which can help to instantiate these variables. The activities on the stack can be represented as shown in Table 1.

4.3 Speech Acts

ADS employed four speech acts to capture the information exchange in a dialogue and we use these acts to fill in the missing pieces of information in the predicate. Each speech act is implemented as a (set of) rule(s), which include the utterance (or action) triggered by that act, the initiator of the act, and a variable (if needed) to be filled by the act.

5 Implementation of the System

The system is implemented using Java 1.5 and Prolog 3.12. Java is used to provide a Graphical User Interface (GUI) between the user and the database from which the hotel reservation information is retrieved. Prolog is used to implement the parser and to represent the speech acts. Prolog can be effectively used to express grammar rules. The effect of the triggered rules is made visible to the user using the Java-Prolog Language interface Library (JPL) which is provided as a separate tool with SWI-Prolog. The database which holds the hotel information is built using Microsoft Access 2007 and is accessed by the DM using the Structured Query Language (SQL) library provided by Java. The tools used in the implementation are shown in Fig. 6.

5.1 Implementation of the Parser

We have opted to employ SICStus Prolog 3.12.2 to implement the parser because Prolog can be effectively used in natural language analysis as (1) Prolog is a logic programming language which employs logic to express grammar rules and (2) we were not aiming at testing the efficacy of the Arabic grammar base.

Once the grammar rules are compiled into Prolog, they receive a procedural interpretation, becoming a top-down left-to-right recursive-descent parser. That is to say, by representing the rules of grammar as axioms in Prolog horn-clause logic, we can use Prolog theorem proving engine

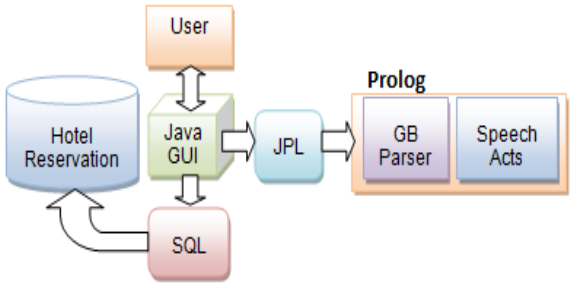

Fig 6. The tools used in implementing ADS

as a parser. As mentioned above, the input is a sequence of tags. If there is a match, the system points out the possible syntax structures upon which the input sentence is built. However, if there is no match, the system returns 'No' to indicate the mismatch with sentence rules. Fig.2 shows the parse tree of the user input اريد حجز غرفة مفردة, *oreed-u hajz-a ghorfah mofradah* "I want to reserve a single room".

5.2 Interfaces of the System

5.2.1 Graphical User Interface

The GUI is implemented in Java. It has been used because it provides a user friendly interface to users through the "swing" library, which includes dialog boxes and drop-down lists.

5.2.2 Database Interface

The SQL library provided with Java 1.5 helps in providing a connection to the database engine (we used MS Access 2007) where data about the hotel (room numbers, prices, etc.) is kept.

The SQL queries are built accumulatively by triggering a series of speech acts and checking with IS. Table 3 shows how an SQL query Q is built after U3 is uttered:

Q = SELECT * FROM rooms
WHERE r_type = 'single' AND date_available BETWEEN 15/12/2014' AND '15/12/2014'+1.

After the production of utterance U4 in which the user confirms the reservation, Q is used to trigger a change of the date_available field to

Table 3. Execution of the SQL query Q

S1	أهلا بك في فندقنا، كيف يمكنني أن أساعدك؟ ahla-n bika fi *fondoki-na, kaif-a uomkinuni ?an osa‡eduk-a?* Welcome to our hotel, how can I help you?
U1	أريد حجز غرفة مفردة *Oreed-u hajz-a ghorfah mofradah* I want to reserve a single room.
S2	كم يوما سيكون الحجز؟ *km yawm-n sa-yakoon alhajez?* How many days is the reservation for?
U2	ليوم واحد, *le-yawm-n wahed* For one day.
S3	في أي تاريخ سيكون الحجز؟ *fi ay tareekh sayakoon alhajej?* For which date is the reservation?
U3	15/12/2014
S4	تتوفر غرفة في هذا التاريخ، هل تريد تأكيد الحجز؟ *ta-tawafer ghorfah fi hatha at-tareekh, hal toreed-u ta?keed-a alhajej?* There is a room available, do you want to confirm the reservation?
U4	نعم, *na3m* "Yes"
S5	تم الحجز، حسابك هو48 دينار *tam alhajez-u, hesabuk-a huw-a 48 dinar* Reservation confirmed, your balance is 48 JDs.
S6	هل لديك استفسارات أخرى؟ *hal ladaik-a estifsarat-n ukhra?* Do you have any other questions?
U5	لا, *la* "No"
S7	شكرا لاستخدامك خدمات فندقنا *Shukran le-estekhdamik-a khadamat-i fondokina* Thanks for using our hotel services.

15/12/2014 (as the user wants to reserve the room for only one day) making it unavailable for reservation by other users on this particular date:

```
UPDATE rooms rr
SET date_available = date_available +1.
```

5.2.3 Prolog Interface

JPL is a set of Java classes providing an interface between Java and Prolog. JPL uses the Java Native Interface (JNI) to connect to the Prolog engine through the Prolog Foreign Language Interface (FLI). JPL only supports the embedding of the Prolog engine within the Java VM. JPL is designed in two layers, a low-level interface to the Prolog FLI and a high-level Java interface for the Java programmer who is not concerned with the details of the Prolog FLI. We use JPL to access the rules in Prolog to send transliterated user input to Prolog to parse the sentence and then receive the parsed text as a Prolog list.

A sample of the dialogue and message boxes of a system's run on Q are shown in Fig. 7 a, b, c.

6 Comparing ADS with other Systems

Before we discuss the methodology of evaluating our system, it is worthwhile to shed light on the ADS components and compare them with some other dialogue systems.

1. As we mentioned earlier in Section 2, we decided to employ a GB-based parser. The most comprehensive work on Arabic parsers, to our knowledge, is presented in [13, 14]. Other approaches to Arabic text parsing employ Lexical Functional Grammar (LFG) [11, 30]. The work of [30] aimed at enriching

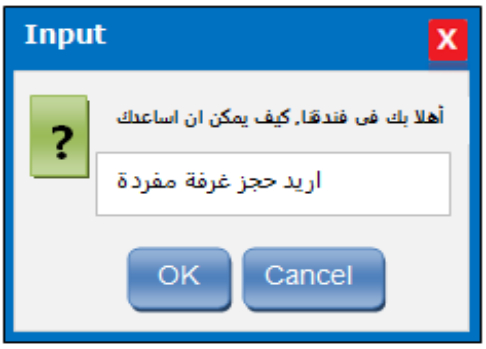

Fig. 7a. A sample "request" dialogue on ADS for a single room reservation

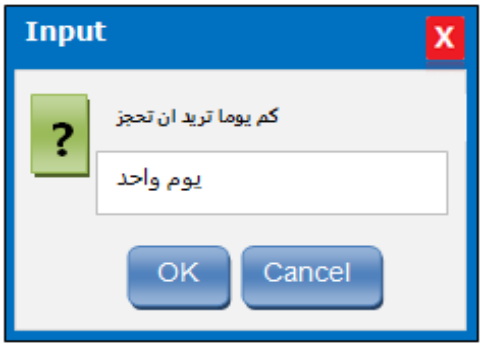

Fig. 7b. A sample "request" & "inform" dialogue on ADS for one day reservation

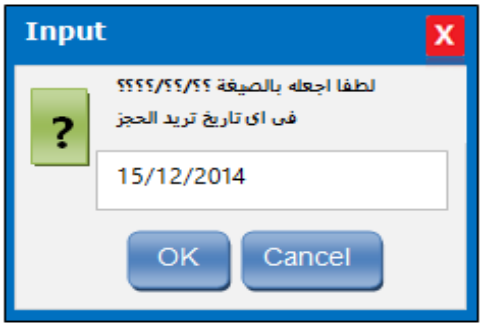

Fig. 7c. A sample "request" & "inform" dialogue on ADS for a date reservation

the Arabic Penn Treebank (ATB) trees and ATB-trained parser output with functional tags.

2. The ADS system uses the "slots-fill" method to generate the appropriate query made by the user. The technique employed by ADS is similar to the Philips train timetable system [2]. However, the Philips system does not keep track of the dialogue acts generated throughout the utterances exchange. In addition, ADS employs an "information-states" stack which observes unfilled slots and interact with the user to fill them in order to form the query.
3. There are many approaches to build a dialogue manager. Cuayáhuitl *et al.* [8] employed the Markov Decision Process (MDP) which uses a stochastic model to associate a proper speech act with an utterance. This may not be appropriate in a dialogue that requires immediate responses. On the contrary, ADS employs information states which are formed as a list of pairs of arguments and theirs corresponding values. Information states help in keeping track of the dialogue moves and also helps when the user initiates sub-dialogues within the current dialogue.
4. Chai *et al*. [7] built an "interpreter" which extracts constraints from a labeled text (output of the parser) describing specifications arranged into chunks of related phrases in order to construct semantic representation trees that are traversed in a bottom-up left-most order. In this regard, ADS has a set of predefined "shallow-trees" which hold nouns as leaves. It then compares a parsed utterance against those leaves in a top-down left-most order, following the language structure generated by the GB-based parser.

7 Testing and Evaluation of the System

In this section we report on the experiments and evaluation of the results produced by ADS.

7.1 Evaluation

The purpose of evaluation is to assess the system to see if it does what it is supposed to do and that everyone is satisfied with it [32]. However, evaluating a dialogue system properly is not an

Table 4. Overall performance based on scores from human evaluators

Q#	# of Evaluators	Very good		Good		Neutral		Bad		Very bad		Overall Performance %
		n	%	*n*	%	*n*	%	*n*	%	*N*	%	
Q-1	500	257	51.4	206	41.2	30	6.0	5	1.0	2	0.4	92.6
Q-2	500	258	51.6	152	30.4	62	12.4	28	5.6	0	0.0	82.0
Q-3	500	354	70.8	106	21.2	11	2.2	28	5.6	1	0.2	92.0
Q-4	500	451	90.2	41	8.2	8	1.6	0	0.0	0	0.0	98.4
Q-5	500	346	69.2	136	27.2	16	3.2	2	0.4	0	0.0	96.4
Overall Satisfaction		**66.6**		**25.6**		5.1		2.5		0.1		**92.3**

n: number of evaluators

easy task, so designers face many complicated issues [18, 32]. One of these issues is the fact that dialogue systems target end users, so usability factors such as satisfaction or likelihood of future use should be the final criteria. Another issue is related to usability factors which are subjective; they can be unpredictable and highly dependent on features of the user interface [17]. According to the handbook of the Expert Advisory Group on Language Engineering Standard (EAGLES), there are 2 methods for evaluating such systems [10]:

1. The "black box" method, which only considers inputs and outputs of the system without having to look inside the system and see how it is built.
2. The "glass box" method, which evaluates the contribution of the system's components to the task at hand.

To evaluate ADS, we have built a database of sentences gathered from hotels' web sites and by interviewing hotels' receptionists in Amman, Jordan. We used the dataset to test if the parser produces correct parsed text to be used by the dialogue manager. In the conducted experiments, we intended to test three aspects as follows.

1. The parser: it checks the grammaticality of input sentences using the GB-based grammar rules and consequently constructs their syntactic structure if they are grammatically valid. In other words, when an input sentence is not grammatically valid, the parser does not produce a syntactic structure and the dialogue manager asks the user to rephrase the utterance.
2. Semantic representation: to ensure that the system understands the user intention, we asked the users to employ different variations of the same sentence to check if the system can handle these variations. For example, the user may say اريد حجز غرفة مفردة, *oreed-u hajza ghorfah mofradah* "I want to reserve a single room" or هل تتوفر غرفة مفردة؟, *hal ta-tawafar-u ghorfah mofradah* "Is there a single room available?" The system was able to understand input in all variations.
3. The system responses: stacking the information states in the system and retrieving or updating data in the database triggers responses to the user input. These responses are built in the system as a part of the information states. Users who interacted with the system were satisfied with the responses generated by the system.

7.2 Evaluation Methodology

The final global evaluation of ADS focuses on the black box method to assess the informativeness of the dialogues produced by the system based on human evaluators. To evaluate ADS, we

adapted some of the Trindi's[1] wish list of desired dialogue behavior, which is specified as a "tick-list'" questions. The metric we used is based on assigning a score between 0-4 (cf. Section 7.4) to reflect the satisfaction of the evaluators on the behavior of ADS. We used a questionnaire of five basic questions as follows.

Q1) Could the system deal with answers to questions that give more information than was requested?
Q2) Could the system deal with answers to questions that give information different from that was actually requested?
Q3) Could the system deal with answers to questions that give less information than was actually requested?
Q4) Could the system deal with sub-dialogues initiated by the user?
Q5) Did the system ask only appropriate follow-up questions?

7.3 Human Evaluators

To determine the effectiveness of ADS, we determined to evaluate the dialogues using human subjects. We contacted our colleagues at King Abdullah II School for Information Technology (KASIT) at the University of Jordan and informed them about our system and the experiments we needed to run. We asked them to motivate their students to participate. All our findings are backed by the analysis of these experiments that we performed with the help of human evaluators. We assumed that the evaluators had good reading and comprehension skills in Arabic and they varied in their educational levels. Finally, we had 500 participants whose majors included computer science, computer information systems, and business information technology.

7.4 Experiments

The experiments were conducted in one of the KASIT's computer labs. The lab has 40 desktop machines and ADS was installed in the lab. To collect the results, we attended around 15 sessions of 30-40 students each. After a short presentation about ADS and how it runs, we asked the evaluators to interact and read carefully through the dialogue and then to provide a score between 0-4 representing their satisfaction with the dialogues and the acts produced by ADS based on the following scale: 0=*very bad*, 1=*bad*, 2=*neutral*, 3=*good* and 4=*very good*. Descriptive statistics (percentages) were computed for the analysis of data and interpretation of results.

7.5 Results and Discussion

The detailed results obtained from the evaluators for their judgments on the five basic questions are presented in Table 4. Before discussing the results, we can draw the following three conclusions:

1. The system works at least sometimes.
2. If we count a dialogue as successful when the human evaluators evaluate it as *very good* only, then the overall satisfaction of the system is 66.6%.
3. If we count a dialogue as successful when the human evaluators evaluate it as *good or very good*, then the overall satisfaction of the system is 92.3%.
4. Fig.8 shows the overall satisfaction of the human evaluators experimenting with ADS.

Now we will discuss the behavior of ADS in terms of each of the five questions presented above.

Concerning the first question which tests whether the system can deal with answers to questions that give more information than was requested, ADS performance was good. This can be explained by the fact that ADS deals with every new utterance as a new dialogue before pushing it into the information states stack. In other words, ADS uses every available piece of information in the utterance to fulfill the ultimate goal of the dialogue.

With respect to the second question, ADS scores were not very high because the system considers utterances with information which differs from the expected one as new sub-dialogues with new goals. Speech acts associated with an utterance has a pre-defined

[1] Trindi http://www.ling.gu.se/projekt/trindi

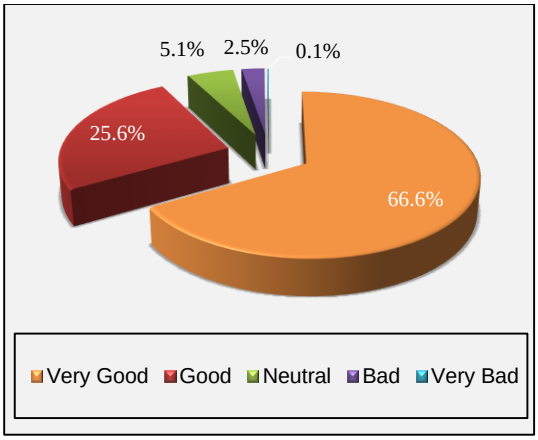

Fig. 8. Overall satisfaction of evaluators experimenting with ADS

argument to fill, which may not match well with the intended goal of the dialogue.

ADS performance with respect to the third question, which tests whether the system can deal with answers to questions that give less information than was actually requested, was also good. ADS can deal with such cases because when an argument associated with the speech act in an utterance is not filled, it asks the user to provide the missing information.

Concerning the fourth question, ADS performance was good. This is due to the fact that ADS keeps track of the dialogue moves in an agenda (implemented as a stack). Hence, it can deal with sub-dialogues by setting new goals which, when fulfilled, get the system back to the main dialogue sequence in order to satisfy the main goal.

Finally, concerning the last question, which tests whether the system asks only appropriate follow-up questions, ADS performance was good. This can be explained by the fact that ADS has a sequence of questions to fill in missing information for the goal set at the beginning of the dialogue and does not object to interruptions by the user.

When analyzing the way some users interact with the system, we found that they have weaknesses in writing Arabic. Overall, we are convinced that there is a need for more testing and comparison with human operated systems.

8 Conclusions and Future Work

In this paper, we made a step toward developing an Arabic Dialogue System (ADS) that allows a user to make reservations in a hotel using written Arabic text. We had to limit the scope of the parser (cf. Section 4.1) so that it analyzes the syntactic structure of some simple Arabic sentences' structures based on the GB Theory. We have presented an implementation of the grammar rules in Prolog. We intend to compare ADS with similar systems. We hope to enhance the system (1) by using a morphological analyzer that would provide important features about the words such as clitics, number, and gender, (2) by adding more rules to include more sentence structures, and (3) by including other syntactic features such as subject-verb agreement on number and gender, words clitics, and cases represented as suffixes to nouns.

Keeping the history of a dialogue is important in tracking dialogue acts in the system. We have provided an implementation of the agenda that keeps track of the dialogue acts when the system is put into use. It may be necessary to investigate the possibility of generating contextually sensitive answers. We believe that user satisfaction would be enhanced if the system could provide and negotiate more informative and/or helpful answers. We have shown by experiments that the results we obtained confirm the viability of using ADS to tackle the problem of interactive Arabic dialogues.

References

1. **Allen, J., Byron, D., Dzikovska, M., Ferguson, G., Galescu, L., & Stent, A. (2000).** An architecture for a generic dialogue shell. *Natural Language Engineering,* Vol. 6, No. 34, pp. 213–228.
2. **Aust, H., Oerder, M., Seide, F., & Steinbiss, V. (1995).** A spoken language inquiry system for automatic train timetable information. *Philips Journal of Research*, Vol. 49, No. 4, pp. 399–418.
3. **Black, C. (1999).** *A step-by-step introduction to the government and binding theory of syntax*. SIL - Mexico Branch and University of North Dakota.

4. **Bohus, D. & Rudnicky, A. (2009).** The RavenClaw dialog management framework: Architecture and systems. *Computer Speech and Language,* Vol. 23, No. 3, pp. 332–361.
5. **Boye, J. (2007).** Dialogue management for automatic troubleshooting and other problem-solving applications. *Proceeding of the 8th SIGdial Workshop on Discourse and Dialogue*, pp. 247–255.
6. **Btoosh, M. (2010).** Wh-movement in standard Arabic: an optimality-theoretic account. *Poznan Studies in Contemporary Linguistics*, Vol. 46, No.1, pp. 1–26.
7. **Chai, J., Horvath, V., Nicolov, N., Stys, M., Kambhatla, N., Zadrozny, W., & Melville, P. (2002).** Natural language assistant: A dialog system for online product recommendation. *AI Magazine*, Vol. 23, No. 2, pp. 63–75.
8. **Cuayáhuitl, H., Dethlefs, N., Richter, K.F., Tenbrink, T., & Bateman, J. (2010).** A dialogue system for indoor wayfinding using text-based natural language. *Journal of Computational Linguistics and Applications*, Vol. 1, pp. 285–304.
9. **Fraser, N. (1995).** Messy data, what can we learn from it? *Proc. of 9th Twente workshop on Language technology: Corpus-based approaches to dialogue modelling*, pp. 95–105.
10. **Gibbon, D., Moore, R., & Winski, R. (Eds.) (1997).** Handbook of standards and resources for spoken language systems. *Spoken Language System Assessment*, 3, Mouton De Gruyter.
11. **Habash, N., Reem, F., & Ryan, R. (2009).** Syntactic annotation in the Columbia Arabic treebank. *Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR)*, Egypt, pp. 125–132.
12. **Haegeman, L. (1991).** *Introduction to government and binding theory*, Blackwell.
13. **Hammo, B., Moubaiddin, A., Obeid, N., & Tuffaha, A. (2014).** Understanding Arabic syntactic structure in light of the government and binding theory. *Proceedings of CICLing 2014, 15th International Conference on Intelligent Text Processing and Computational Linguistics*, Kathmandu, Nepal.
14. **Hammo, B., Moubaiddin, A., Obeid, N., & Tuffaha, A. (2014).** Formal description of Arabic syntactic structure in the framework of the government and binding theory. *Computación y Sistemas*, Vol. 18, No. 3, pp. 611–625.
15. **Homeidi, M. (2003).** The notion of governor in modern standard Arabic (MSA) and English: A contrastive perspective. *J. King Saud Univ*., Vol. 15, pp. 49–62.
16. **Hulstijn, J., Steetskamp, R., ter Doest, H.W.L., van de Burgt, S.P., & Nijholt, A. (1996).** Topics in SCHISMA dialogues. *Proceedings of the Twente Workshop on Language Technology: Dialogue Management in Natural Language Systems*, pp. 89–99.
17. **Jurafsky, D. & Martin, J. H. (2009).** *Speech and language processing* (*Chapter 19, Dialogue and Conversational Agents*). Pearson International Edition.
18. **Kamm, C., Walker, M.A., & Litman, D. (1999).** Evaluating spoken language systems. *Proceedings of American Voice Input/Output Society (AVIOS)*.
19. **Komatani, K., Kanda, N., Nakano, M., Nakadai, K., Tsujino, H., Ogata, T., & Okuno, H.G. (2006).** Multi-domain spoken dialogue system with extensibility and robustness against speech recognition errors. *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pp. 9–17.
20. **Lamel, L., Rosset, S., Gauvain, J.L., & Bennacef, S. (1999).** The LIMSI ARISE system for train travel information. *Proceedings of the IEEE international Conference on Acoustics, Speech and Signal Processing*, pp. 501–504.
21. **Larsson, S. & Ericsson, S. (2002).** GoDiS - issue-based dialogue management in a multi-domain, multi-language dialogue system. *Proceedings of the ACL-02 Demonstrations Session*, pp. 104-105.
22. **Lemon, O., Gruenstein, A., & Peters, S. (2002).** Multi-tasking and collaborative activities in dialogue systems. *Proceedings of the SIGDIAL Workshop on Discourse and Dialogue*, pp. 113–124.
23. **Martínez-Miranda, J., Aldea, A., & Bañares-Alcántara, R. (2004).** Agent Based Simulation in the Selection of Work Teams. *Computación y Sistemas*, Vol. 7, No. 3, pp. 210–223.
24. **McTear, M. (1998).** Modelling spoken dialogues with state transition diagrams: Experience of the CSLU toolkit. *Proceedings of the International Conference on Spoken Language Processing*, pp. 1223–1226.
25. **Moubaiddin, A., Tuffaha, A., Hammo, B., & Obeid, N. (2013).** Investigating the syntactic structure of Arabic sentences. *Proceedings of Communications, Signal Processing, and their Applications,* IEEE, pp. 1–6.
26. **Pakucs, B. (2003).** Towards dynamic multi-domain dialogue processing. *Proceedings of the European Conference on Speech, Communication and Technology*, pp. 741-744.

27. **Pellom, B., Ward, W., Hansen, J., Cole, R., Hacioglu, K., Zhang, J., Yu, X., & Pradhan, S. (2001).** University of Colorado dialog systems for travel and navigation. *Proceedings of the first international conference on Human language technology research (HLT),* Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1–6.
28. **Shala, L., Rus, V., & Graesser, A.C. (2010).** Automated speech act classification in Arabic. *Subjetividad y Procesos Cognitivos*, Vol. 14, 284-292.
29. **Sporka, A., Franc, J., & Riccardi, G. (2009).** Can machines call people? User experience while answering telephone calls initiated by machine. *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems (CHI EA '09)*, ACM, New York, NY, USA, pp. 3625–3630.
30. **Tounsi, L. & Genabith, J.V. (2010).** Arabic Parsing Using Grammar Transforms. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC*), Valletta, Malta, pp. 1986–1989.
31. **Walker, M.A., Aberdeen, J.S., Boland, J.E., Bratt, E.O., Garofolo, J.S., Hirschman, L., & Whittaker, S. (2001).** DARPA communicator dialog travel planning systems. *Proceedings of the European Conference on Speech, Communication and Technology*, pp. 1371–1374.
32. **Young, S.R. (1989).** Evaluation techniques for spoken language systems. *ESCA Tutorial and Research Workshop on Speech Input/Output Assessment and Speech Databases*, Vol. 2, pp. 219–222.
33. **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 Audio, Speech, and Language Processing*, Vol. 8, No. 1, pp. 85–96.

Asma Moubaiddin received her B.A. in Language and Literature from the University of Jordan, Amman, Jordan, in 1975. She received her M.A. in Applied Linguistics in 1986 and Ph.D. in Language and Linguistics from University of Essex, England, U.K., in 1992. She is an Assistant Professor at the Department of Linguistics and she has been with the University of Jordan since 2004. Her research interests include syntax, semantic, language acquisition, knowledge sharing, and dialog systems.

Ola Shalbak received her B.Sc. in Computer Science in 2004 from the University of Jordan, Amman, Jordan. She obtained her M.Sc. in Computer Information Systems from the University of Jordan, Amman, Jordan, in 2012. She has worked as a lab supervisor and software engineering trainer at the University of Jordan since 2006. Her research interests include Arabic dialogue systems, syntax, and semantics.

Bassam Hammo received his B.Sc. in Computer Science from the University of Jordan, Amman, Jordan, in 1987. He has M.Sc. in Computer Science from Northeastern University and Ph.D. in Computer Science from DePaul University, Chicago, IL, USA (2002). He is an Associate Professor of NLP at the Department of Computer Information Systems and has been with the University of Jordan since 2003. His research interest is Arabic natural language processing and its applications. He leads the ANLP research group at the University of Jordan.

Nadim Obeid received his B.Sc. in Mathematics and B.Sc. in Business Administration from Lebanese University, Lebanon, in 1979 and 1980, respectively. He has M.Sc. in Computer Studies and obtained Ph.D. in Computer Science from University of Essex, England, U.K., in 1987. He is a full time Professor at the Department of Computer Information Systems and has been with the University of Jordan since 2004. His research interests include knowledge representation, multi-agents, and dialog systems.

Article received on 14/04/2014; accepted on 23/01/2015.
Corresponding author is Asma Moubaiddin.