

This document is published in:

Journal of Ambient Intelligence and Smart Environments (2012), 4 (3), 183-207.

DOI: <http://www.dx.doi.org/10.3233/AIS-2012-0145>

© 2012 IOS Press

Bringing together commercial and academic perspectives for the development of intelligent AmI interfaces

David Griol ^a, José Manuel Molina ^a and Zoraida Callejas ^b

^a *Department of Computer Science, Carlos III University of Madrid, Avda. de la Universidad, 30, 28911. Leganés, Spain*

E-mail: {david.griol,josemanuel.molina}@uc3m.es

^b *Department of Languages and Computer Systems, University of Granada, CITIC-UGR, C/ Pda. Daniel Saucedo Aranda s/n, 18071. Granada, Spain*

E-mail: zoraida@ugr.es

Abstract: The users of Ambient Intelligence systems expect an intelligent behavior from their environment, receiving adapted and easily accessible services and functionality. This can only be possible if the communication between the user and the system is carried out through an interface that is simple (i.e. which does not have a steep learning curve), fluid (i.e. the communication takes place rapidly and effectively), and robust (i.e. the system understands the user correctly). Natural language interfaces such as dialog systems combine the previous three requisites, as they are based on a spoken conversation between the user and the system that resembles human communication. The current industrial development of commercial dialog systems deploys robust interfaces in strictly defined application domains. However, commercial systems have not yet adopted the new perspective proposed in the academic settings, which would allow straightforward adaptation of these interfaces to various application domains. This would be highly beneficial for their use in AmI settings as the same interface could be used in varying environments. In this paper, we propose a new approach to bridge the gap between the academic and industrial perspectives in order to develop dialog systems using an academic paradigm while employing the industrial standards, which makes it possible to obtain new generation interfaces without the need for changing the already existing commercial infrastructures. Our proposal has been evaluated with the successful development of a real dialog system that follows our proposed approach to manage dialog and generates code compliant with the industry-wide standard VoiceXML.

Keywords: Dialog systems, statistical methodologies, VoiceXML, web interfaces, speech interaction

1. Introduction

Ambient Intelligence (AmI) systems and Smart Environments (SmE) build on three key technologies: ubiquitous computing, ubiquitous communication and intelligent user interfaces [5,36]. Thus, they usually consist of a set of interconnected computing and sensing devices that surround the users pervasively in their environment and are invisible to them, providing ser-

vices that are dynamically adapted to the interaction context and easily accessed through the interface [4].

In order for the user to perceive the system as intelligent, it is necessary to ensure an effective, easy, safe and transparent interaction between the user and the system. With this objective, as an attempt to enhance and ease human-to-computer interaction, in the last years there has been an increasing interest in simulating human-to-human communication by means of the so-called spoken dialog systems (SDS) [21,33,37,40]. A dialog system can be defined as software that accepts natural language as input and generates natural

language as output, engaging in a conversation with the user. As speech is a common, spontaneous and simple mean of communication [12], people opt to employ their voice to access information, and control AmI applications. Even when there is a possibility to choose between different modalities, several studies have shown that oral communication is usually preferred [7].

One of the core aspects of developing dialog systems for AmI applications is to design flexible dialog management strategies. The dialog strategy defines the system conversational behavior in response to user utterances and environmental states that, for example, can be based on observed or inferred events or beliefs. This is the fundamental task of dialog management [46], as the performance of the system depends to a great extent on the quality of this strategy. However, there is no clear definition of what constitutes a good strategy [74], and thus a great effort is employed in commercial systems to design them and find empirical evidence of their appropriateness. This design is usually carried out in industry by hand-crafting dialog strategies tightly coupled to the application domain in order to optimize the behavior of the dialog system in that context. However, it is a very time-consuming process and has the disadvantage of lack of portability and adaptation to new contexts.

This has motivated the research community to find ways for automating dialog learning by using statistical models trained with real conversations. Statistical approaches can model the variability in user behaviors and allow exploring a wider range of strategies. Although the construction and parameterization of the model depends on expert knowledge of the task, the final objective is to develop dialog systems that have a more robust behavior, better portability, and are easier to adapt to different user profiles or tasks.

As the success of statistical approaches depends on the quality of the data used to develop the dialog model, considerable effort is necessary to acquire and label a corpus with the data necessary to train a good model. In order to mitigate this, user simulators appeared as an efficient means to generate dialogs between the system and a simulated user [59]. This way, the user simulator makes it possible to generate a large number of dialogs in a very simple way, therefore reducing the time and effort required for the evaluation of a dialog system, as well as allowing to evaluate it in early development phases.

Unfortunately, dialog systems in industry have been evolving on a parallel path with those in academic re-

search. This way, commercial systems are being deployed using industry-wide standards and protocols such as VoiceXML¹, for which different programming environments and tools have been created to help developers. These programming standards allow the definition of a dialog strategy based on scripted Finite State Machines. However, the application of statistical approaches to dialog management makes it possible to consider a wider space of dialog strategies [23,29,72], thus allowing to create dynamic and adapted dialogs, which are easier to adapt to different user profiles and application domains than the industry rule-based approaches.

Recently some authors have started to point out the need for a “synergistic convergence” of architectures, abstractions and methods from both communities, so that the interesting ideas and technologies that have appeared in academic research are not overlooked by industry practitioners [1,44,46–48].

This paper endeavors to bridge the gap between the two communities so that research results can be brought to commercial systems and benefit current users of AmI applications. As an attempt to improve the current technology, we propose to combine the flexibility of statistical dialog management with the facilities that VoiceXML offers, thus introducing statistical methodologies for the development of commercial (and not strictly academic) dialog systems. Our technique employs a statistical model based on neural networks that takes into account the history of the dialog up to the current dialog state in order to predict the next system response. The dialog model is learned from a labeled training corpus for the task and is mainly based on modeling sequences of the system and user dialog acts and the definition of a data structure which takes into account the data supplied by the user throughout the dialog, and makes the estimation of the model from the training data manageable. Our statistical dialog management technique has been previously applied to several tasks (including e-health [28], a fast food domain [35], or travel planning [27]) to verify its correct operation in very well-known domains related to commercial applications of dialog systems.

This paper extends our previous work by making possible to exploit expert knowledge about deployment of VoiceXML applications, as well as current development environments and tools, with the advantage that transitions between dialog states are carried

¹<http://www.w3.org/TR/voicexml20/>

out on a data-driven basis (i.e., it is not a deterministic process). In addition, the system prompts and the grammars for automatic speech recognition are implemented in VoiceXML-compliant formats (e.g., Java Speech Grammar Format or JSFG, and Speech Recognition Grammar Specification or SRGS). We also provide an automatic dialog simulation technique as a solution to acquire the data that is required to learn the dialog model and construct the speech grammars to then complete the VoiceXML application.

The remainder of the paper is organized as follows. Section 2 describes the related work about industrial and research perspectives of the development of spoken dialog systems. This section is focused on key aspects related to our proposal, such as design practices, objectives and current trends, architectures, dialog management techniques, standards, application of statistical methodologies, and user modeling. Section 3 describes our proposal to develop a spoken dialog system based on a statistical methodology for dialog management, a dialog generation technique to automatically acquire the data that is required to learn the dialog model and speech grammars required for the dialog system, and the use of the VoiceXML standard to carry out the implementation of the system. Section 4 presents a detailed explanation of how our proposal has been applied to develop a practical dialog system that works as an academic assistant. Section 5 presents the criteria defined to carry out the evaluation of the developed dialog system, whereas Section 6 discusses the evaluation results. Finally, our conclusions and future work are presented in Section 7.

2. Related work

The spoken dialog industry has reached a maturity based on standards that pervade technology to provide high interoperability, which makes it possible to divide the market in a vertical structure of technology vendors, platform integrators, application developers, and hosting companies [48].

The design practices of conventional commercial dialog systems are currently well established in industry. In these practices, voice user interface (VUI) experts [3] handcraft a detailed dialog plan based on their knowledge about the specific task and the business rules (e.g., to verify the user's identity before providing certain information). In addition, designers commonly define the precise wording for the system prompts according to the dialog state and context, and also the ex-

pected types of user's utterances for each turn. As described in [47,73], this approach is well-documented [13] and has been used to develop hundreds of successful commercial dialog systems.

This standard procedure to develop commercial dialog systems can be represented as a graph that describes the set of dialog states and tables containing the details of each state. Transitions between dialog states are determined by the user turns and the result of different system operations (e.g., the results of database queries).

The main objectives of this approach are usability and task completion, and the main challenge is to cope with the limitations of the front-end technology, as speech recognition errors are common. To solve this situation, especially when the domain model is quite simple and known by the users, commercial applications are usually implemented as directed dialogs, in which users are restricted and guided to provide specific pieces of information, thus restricting the possible user responses and minimizing the probability of speech recognition errors. This way, each interaction is designed to accept a restricted set of expected user reactions to the specific prompt played at that particular turn, providing the speech recognizer with an appropriately designed grammar with a small list of synonyms (thus, generic system prompts like "how can I help you today?" are seldom used).

This paradigm has facilitated the development of standards that are governing the speech industry, such as VoiceXML, SRGS (Speech Recognition Grammar Specification²), SSML (Speech Synthesis Markup Language³), SISR (Semantic Interpretation for Speech Recognition⁴), and CCXML (Voice Browser Call Control⁵). VoiceXML allows creating dialog systems that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input, recording of spoken input, telephony, and mixed initiative conversations. Its major goals are to bring the advantages of web-based development and content delivery to interactive voice response applications, and to free the authors of such applications from low-level programming and resource management. It enables integration of voice services with data services using the familiar client-server paradigm.

²<http://www.w3.org/TR/speech-grammar/>

³<http://www.w3.org/TR/speech-synthesis/>

⁴<http://www.w3.org/TR/semantic-interpretation/>

⁵<http://www.w3.org/TR/ccxml/>

Current trends of the industry of spoken dialog systems today also include the development of reusable components and prepackaged applications, reducing deployment costs and risks and, at the same time, simplifying the design and development of more sophisticated applications [1,44,46,48].

On the other side, spoken dialog research has been moving on a parallel path trying to attain naturalness and freedom of communication. This way, current research lines include very important topics that can be classified into the following three categories:

- Understanding human-human communication and the differences with human-computer interaction (HCI): understand the mechanisms of human dialog through linguistically motivated studies on human-human corpora.
- Designing interfaces for usable systems: develop general design principles that, once applied, would result in usable human-machine user interfaces based on speech recognition and speech synthesis technology.
- Developing these usable systems: formalize programming styles, models, engines and tools which can be used to build effective dialog applications.

In the literature we can find different systems and research projects focused on the integration of speech interaction and dialog systems in AmI and SmE. As described in [21,34,40], current research areas and future trends are focused on assistive, adaptive and proactive system design, dialog management and system-environment interaction. This is the case of the Homey project [39], for which an intelligent dialog interface was designed to develop a dialog with dynamic adaptation between a tele-medicine interface and a patient. Saini et al. [54] report an exploration of the concept of social intelligence in the context of home dialog systems for an intelligent home, concluding that endowing a home dialog system with social intelligence may create a positive bias in user's perception of technology in the environment, increase user acceptance for the home dialog system, and trigger social behaviors of the user towards the home dialog system. Montoro et al. [43] focus on the interpretation and generation processes of an interface for AmI systems. In their architecture, the interface is automatically created for each specific environment and the interpretation and generation vary depending on the environment and its context. Espejo et al. [19] describe the implementation of the Mayordomo multimodal dialog system, which includes Radio Frequency IDentification (RFID) devices

to locate users and adapt the dialog interaction to centralize the control of the appliances in a home AmI environment.

In the following subsections we discuss the main differences between commercial and academic perspectives for the development of conversational interfaces: architecture, design, dialog management abstractions, and system evaluation.

2.1. Different approaches to architectural design

To successfully manage the interaction with the users, spoken dialog systems usually carry out five tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialog management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules, as shown in Fig. 1.

Speech recognition is the process of obtaining the text string corresponding to an acoustic input. It is a very complex task as there is much variability in the input characteristics, which can differ depending on the linguistics of the utterance, the speaker, the interaction context and the transmission channel. Linguistic variability involves differences in phonetic, syntactic and semantic components that affect the voice signal. Inter-speaker variability refers to the big difference between speakers regarding their speaking style, voice, age, gender or nationality. Furthermore even the same person does not always pronounce the same words in the same way, as people are affected by physical and psychological factors that are highly variable and usually not predictable.

Once the SDS has recognized what the user uttered, it is necessary to understand what he said. Natural language processing is the process of obtaining the semantic of a text string. It generally involves morphological, lexical, syntactical, semantic, discourse and pragmatic knowledge. In a first stage lexical and morphological knowledge allow dividing the words in their constituents distinguishing lexemes and morphemes: lexemes are the part of the words that indicates their semantic and the morphemes are the different infixes and suffixes that allow obtaining different word classes. Syntactic analysis yields a hierarchical structure of the sentences, however in spoken language frequently phrases are affected by the difficulties that are associated to the so-called disfluency phenomena: filled pauses, repetitions, syntactic incompleteness and repairs.

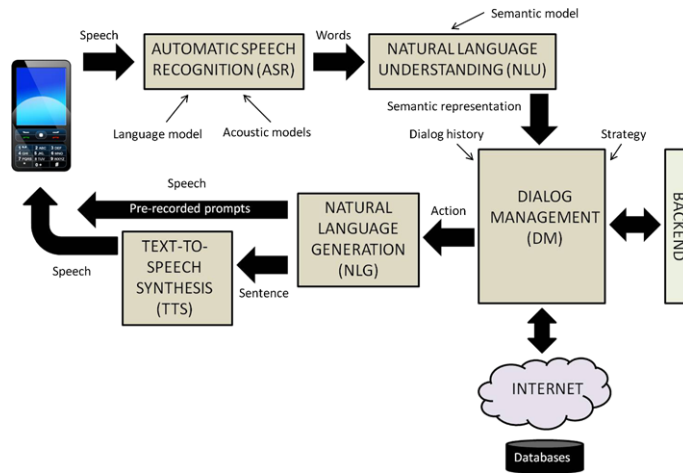


Fig. 1. Modular architecture of spoken dialog systems.

Semantic analysis extracts the meaning of a complex syntactic structure from the meaning of its constituents. In the pragmatic and discourse processing stage, the sentences are interpreted in the context of the whole dialog.

There is not a universally agreed upon definition of the tasks that a dialog manager has to carry. Traum and Larsson [67] state that dialog managing involves four main tasks: i) updating the dialog context, ii) providing a context for interpretations, iii) coordinating other modules and iv) deciding the information to convey and when to do it. Thus, the dialog manager has to deal with different sources of information such as the NLU results, database queries results, application domain knowledge, and knowledge about the users and the previous dialog history. Its complexity depends on the task and the dialog flexibility and initiative. Given that speech recognition is not perfect, one of the most critical operations of the design of the dialog manager is related to error handling. Speech recognition makes errors, language understanding makes errors, and so user interfaces are far from being as effective as humans. For all of this technology to work, one has to impose severe limitations on the scope of the applications, which require a great amount of manual work for the designers. One common way to alleviate errors is to use techniques aimed at establishing a confidence level for the speech recognition result, and to use that for deciding when to ask the user for confirmation, or reject the hypothesis completely and re-prompt the user. Too many confirmations as well as too many reprompts would annoy users. So, it is important to reduce the number of confirmations and rejections to a minimum that also preserves a reasonable level of accuracy.

Natural language generation is the process of obtaining texts in natural language from a non-linguistic representation. It is usually carried out in five steps: content organization, content distribution in sentences, lexicalization, generation of referential expressions and linguistic realization. It is important to obtain legible messages, optimizing the text using referring expressions and linking words and adapting the vocabulary and the complexity of the syntactic structures to the user's linguistic expertise. The simplest approach consists in using predefined text messages (e.g. error messages and warnings). Finally, a text-to-speech synthesizer is used to generate the voice signal that will be transmitted to the user.

Classical architectures defined within the research community include valuable examples like Galaxy [51], used as a testbed for the research and development of several dialog systems in different domains (e.g., automobile classified ads [24], restaurant guide [70], and weather information [25]), different languages [69], and different access mechanisms [24, 25, 70]. This architecture, created as a reference for the DARPA Communicator Program, is based on the client-server paradigm and the introduction of a "hub" to receive and transmit the messages generated by the different modules of the classical architecture described in Fig. 1.

Commercial system architectures evolved in a different way. Early commercial dialog systems were built using proprietary architectures based on IVR (Interactive Voice Response) platforms. Later, the convergence between web technologies and speech applications thanks to the VoiceXML standard has made pos-

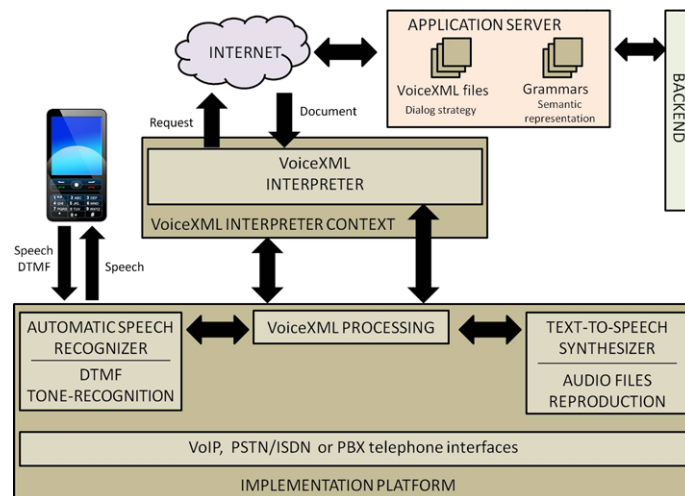


Fig. 2. Typical architecture of commercial dialog systems.

sible the development of standard web architectures (the so called Voice Web). This typical architecture is shown in Fig. 2. As it can be seen, the architecture includes three main components: the application server, the VoiceXML interpreter context, and the implementation platform. A document server (e.g. a web server) processes requests from the VoiceXML Interpreter, through the VoiceXML interpreter context. As a result of these requests, the document server returns VoiceXML documents (including next system actions) and grammars in VoiceXML-compliant formats (used to generate the semantic representation of users' utterances) that are then processed by the VoiceXML interpreter. The interpreter context provides all supported functions that are necessary for the interpreter. The implementation platform is controlled by the VoiceXML interpreter context and by the VoiceXML interpreter. This platform typically integrates speech recognition, speech synthesis, VoiceXML processing, media playback, and required infrastructures for connecting to the Public Switched Telephone Network (PSTN) or VoIP into a complete package for deployment in an IVR. This way, a voice service is viewed as a sequence of interaction dialogs between a user and an implementation platform. Document servers maintain overall service logic, perform database and legacy system operations, and provide these dialogs.

VoiceXML documents instruct the browser to activate the speech resources (speech recognition, TTS, prompt player, etc.) with a specific set of parameters (e.g., a particular grammar for the speech recognition engine, a prompt to be synthesized by the text-to-speech system, or an audio recording to be played).

Once user's speech has been recognized, and the recognition results returned to the browser in the form of a structured set of variables, the browser sends them back to the web server, together with the request of another VoiceXML document. The Web server then replies by sending the requested document to the browser, and the interaction continues following this process.

In summary, the research community has focused on creating new architectures while commercial applications currently make use of a well defined architecture to develop dialog systems and take the advantages of web technologies. Our approach exploits the current technologies and available standards that have facilitated the development of dialog systems, and employ them to build interfaces that follow the new research advances.

2.2. Dialog management approaches

The previous description of the development cycle of commercial SDSs gives an idea of its complexity and cost. Although dialog management is only a part of it, it can be considered one of the most expensive tasks given that this module encapsulates the logic of the speech application. Firstly, developers collect the requirements that describe what the system functional and not functional requirements. In fact, dialog management relies on the fundamental task of deciding what action or response a system should take in response to user input. The selection of a certain action depends on multiple factors, such as the output of the speech recognizer (e.g., measures that define the reli-

ability of the recognized information), the dialog interaction (e.g., the number of repairs carried out so far), the application domain (e.g., guidelines for customer service), and the responses and status of external back-ends, devices, and data repositories. Given that the actions of the system directly impact users, the dialog manager is largely responsible for user satisfaction. This way, the design of an appropriate dialog management mechanism is at the core of dialog system engineering.

The simplest dialog management strategy is programmatic dialog management, in which a generic program implements the application with an interaction model based on finite-state machines [3]. The user actions determine the transitions between the system responses, which are the nodes of the finite-state machine. Users' actions represent the user responses to the system prompts, which are coded in recognition grammars.

These early applications only supported strict directed dialog interaction, in which at each turn the system directs the user by proposing a small number of choices, which also result in a limited grammar or vocabulary at each turn. Directed dialog was efficient in terms of accuracy and cost of development. Although libraries and dialog modules are generally created in the form of sample code or templates, which could be reused and adapted to different applications, the weakest point of this approach is its lack of versatility [1,48].

Unlike the finite-state approach, frame-based dialog managers do not have a predefined dialog path but use a frame structure comprised of one slot per piece of information that the system can gather from the user [37]. The core idea is that humans communicate to achieve goals and during the interaction the mental state of the speakers may change. Thus, frame-based dialog managers model dialog as a cooperation between the user and the system to reach common goals. Utterances are not considered text strings, but dialog acts in which the user communicates his intentions. In this approach, the system interprets speech in order to acquire enough information to perform a specific action. Its advantage is that it can capture several data at once and the information can be provided in any order (more than one slot can be filled per dialog turn and in any order). This is the approach used by most current commercial systems. The Form Interpretation Algorithm (FIA), the basis for the VoiceXML standard, is an example of a functional model of frame-based dialog management.

A related approach is to the so-called "information state" dialog theory [67]. The information state of a dialog represents the information needed to uniquely distinguish it from all others. It comprises the accumulated user interventions and previous dialog actions on which the next system response can be based. The information state is also sometimes known as the conversation store, discourse context or mental state. Following the information state theory, the main tasks of the dialog manager are to update the information state based on the observed user actions, and based on them, to select the next system action.

Additionally, when it is necessary to execute and monitor operations in a dynamically changing application domain, an agent-based approach can be employed. The modular agent-based approach to dialog management makes it possible to combine the benefits of different dialog control models, such as finite-state based dialog control and frame-based dialog management [11]. Similarly, it can benefit from alternative dialog management strategies, such as the system-initiative approach and the mixed-initiative approach [68].

Statistical approaches for dialog management present several important advantages. Rather than maintaining a single hypothesis for the dialog state, they maintain a distribution over many hypotheses for the correct dialog state. In addition, statistical methodologies choose actions using an optimization process, in which a developer specifies high-level goals and the optimization works out the detailed dialog plan. Finally, statistical dialog management systems have shown, in research settings, more robustness to speech recognition errors, yielding shorter dialogs with higher task completion rates [72].

Automating dialog management is useful for developing, deploying and re-deploying commercial applications. In fact, the application of machine learning approaches to dialog management strategy design is a rapidly growing research area. Machine-learning approaches to dialog management attempt to learn optimal strategies from corpora of real human-computer dialog data using automated "trial-and-error" methods instead of relying on empirical design principles [74]. The main trend in this area is an increased use of data for improving the performance of the system.

As described in [46], there are three main categories of elements of the spoken dialog interaction where the use of massive amounts of data can potentially improve automation rate, and ultimately, the penetration and acceptance of speech interfaces in the wider

consumer market. They are task-independent behaviors (e.g., error correction and confirmation behavior), task-specific behaviors (e.g., logic associated with certain customer-care practices), and task-interface behaviors (e.g., prompt selection). However, these three categories have in common today the lack of robust guiding principles which are validated by empirical evidence. Thus, rule-based approaches are not suitable in order to exploit the full potential of the available data. Statistical methodologies can be used to tackle this problem while still allowing designers to have enough control to ensure VUI completeness.

The most widespread methodology for machine-learning of dialog strategies consists of modeling human-computer interaction as an optimization problem using Markov Decision Processes (MDP) and reinforcement methods [31,32,64]. The main drawback of this approach is that the large state space of practical spoken dialog systems, makes its direct representation intractable [75]. Partially Observable MDPs (POMDPs) outperform MDP-based dialog strategies since they provide an explicit representation of uncertainty [52]. This enables the dialog manager to avoid and recover from recognition errors by sharing and shifting probability mass between multiple hypotheses of the current dialog state.

Another disadvantage of the POMDP methodology is that the optimization process is free to choose any action at any time. As a result, there is no obvious way to incorporate domain knowledge or constraints such as business rules. In addition, in the worst case spurious actions might be taken with real users, an especially serious concern if POMDP-based systems are going to handle financial or medical transactions. They are also limited to small-scale problems, since the state space would be huge and exact POMDP optimization is again intractable [75].

An approach that scales the POMDP framework for implementing practical spoken dialog systems by the definition of two state spaces is presented in [76]. Approximate algorithms have also been developed to overcome the intractability of exact algorithms but even the most efficient of these techniques such as Point Based Value Iteration (PBVI) cannot scale to the many thousand states required by a statistical dialog manager [71]. Composite Summary Point Based Value Iteration (CSPBVI) has suggested the use of a small summary space for each slot where PBVI policy optimization can be applied. However, policy learning in this technique can only be performed offline, i.e. at design time, because policy training requires an existing

accurate model of user behavior. An alternative technique for online training based on Q-learning is presented in [66], which allows the system to adapt to real users as new dialogs are recorded. This technique does not require any model of user behavior so user simulation techniques are proposed to iteratively learn the dialog model.

Other authors have combined conventional dialog managers with a fully-observable Markov decision process [30,65], or proposed using multiple POMDPs and selecting actions using hand-crafted rules [71]. In [73], the authors combine the robustness of the POMDP with the developer control afforded in conventional approaches: the (conventional) dialog manager and POMDP run in parallel, but the dialog manager is augmented so that it outputs one or more allowed actions at each time-step. The POMDP then chooses the best action from this limited set. Results from a real voice dialer application show that adding the POMDP machinery to a standard dialog system yields a significant improvement. Other interesting approaches for statistical dialog management are based on modeling the system by means of Hidden Markov Models (HMMs) [14] or using Bayesian networks [38,45].

Our methodology for dialog management (Section 3.1) is based on the estimation of a statistical model from the sequences of the system and user dialog acts obtained from a set of training data. The automatic dialog generation technique, described in Section 3.2, facilitates the acquisition and also adaptation of the dialog system to deal with new domains. The next system response is selected by means of a classification process that considers the complete history of the dialog, which is one of the main advantages regarding the previously described statistical methodologies for dialog management. Another main characteristic is the inclusion of a data structure that stores the information provided by the user. The main objective of this structure is to easily encode the complete information related to the task provided by the user during the dialog history, then considering the specific semantics of the task and including this information in the proposed classification process.

2.3. Evaluation and user modeling

Evaluation of spoken dialog systems is a very cumbersome task, given that it requires a thorough plan from the beginning of the design process to cover every possible step of the system in combination with ev-

ery possible user action. This means that the behavior of the application needs to be validated in all possible situations and conditions. In short, VUI completeness entails that all possible combinations of user inputs and conditions need to be anticipated. This way, testing a sophisticated SDS is often incomplete and bugs always seem to appear after deployment given that it is impractical, time-consuming and burdensome to make a SDS explore all different types of actions with real users. Thus, it is very difficult to carry out comparative evaluations of spoken dialog systems even if they have the same application domain, given that success rates must be considered in combination with dialog efficiency and dialog style measures, time and effort required for developers, adaptation to new requirements or tasks, etc.

A technique that has attracted increasing interest in the research community during the last decade is based on the automatic generation of dialogs between the dialog manager and an additional module, called the user simulator, which represents user interactions with the dialog system. The user simulator makes it possible to generate a large number of dialogs in a very simple way. Therefore, this technique reduces the time and effort that would be needed for the evaluation of a dialog system with real users each time the system is modified.

The construction of user models based on statistical methods has provided interesting and well-founded results in recent years and is currently a growing research area. A probabilistic user model can be trained from a corpus of human-computer dialogs to simulate user answers. Therefore, it can be used to learn a dialog strategy by means of its interaction with the dialog manager. In the literature, there are several corpus-based approaches for developing user simulators, learning optimal management strategies, and evaluating the dialog system [15,23,50,62]. A summary of user simulation techniques for reinforcement learning of the dialog strategy can be found in [59].

In [16,17], Eckert, Levin and Pieraccini introduced the use of statistical models to predict the next user action by means of a n -gram model. The proposed model has the advantage of being both statistical and task-independent. Its weak point consists of approximating the complete history of the dialog by a bigram model. In [32], the bigram model is modified by considering only a set of possible user answers following a given system action (the Levin model). Both models have the drawback of considering that every user response depends only on the previous system turn. Therefore, the

simulated user can change objectives continuously or repeat information previously provided.

In [60–63], Scheffler and Young propose a graph-based model. The arcs of the network symbolize actions, and each node represents user decisions (*choice points*). In-depth knowledge of the task and great manual effort are necessary for the specification of all possible dialog paths.

Pietquin, Beaufort and Dutoit combine characteristics of the Scheffler and Young model and Levin model. The main objective is to reduce the manual effort necessary for the construction of the networks [49,50]. A Bayesian network is suggested for user modeling. All model parameters are hand-selected.

Georgila, Henderson and Lemon propose the use of HMMs, defining a more detailed description of the states and considering an extended representation of the history of the dialog [22]. Dialog is described as a sequence of *Information States* [6]. Two different methodologies are described to select the next user action given a history of information states. The first method uses n -grams [16], but with values of n from 2 to 5 to consider a longer history of the dialog. The best results are obtained with 4-grams. The second methodology is based on the use of a linear combination of 290 characteristics to calculate the probability of every action for a specific state.

Cuayáhuatl et al. [14] present a method for dialog simulation based on HMMs in which both user and system behaviors are simulated. Instead of training only a generic HMM model to simulate any type of dialog, the dialogs of an initial corpus are grouped according to the different objectives. A submodel is trained for each one of the objectives, and a bigram model is used to predict the sequence of objectives.

In [56], a new technique for user simulation based on explicit representations of the user goal and the user agenda is presented. The user agenda is a structure that contains the pending user dialog acts that are needed to elicit the information specified in the goal. This model formalizes human-machine dialogs at a semantic level as a sequence of states and dialog acts. An EM-based algorithm is used to estimate optimal parameter values iteratively. In [58], the agenda-based simulator is used to train a statistical POMDP-based dialog manager.

Möller et al. [42] proposed the SpeechEval User Simulation, an integrated approach to statistically model user behavior and predict user satisfaction by letting a user simulation interact directly with the SDS under test. The central knowledge bases for the user simulation are acquired semi-automatically from cor-

pora, from which extract domain grammars to be used in ASR and NLU, domain ontologies, utterance templates for understanding and for template-based generation, as well as learning dialog act classifiers.

Our approach to develop a user simulator and automatically acquire a dialog corpus (Section 3.2) only requires the definition of the semantics for the task for both the user simulator and the dialog manager, so a previous dialog corpus is not required. The user and the dialog models are iteratively built from the interaction of both modules. The labeled corpus for the specific task is automatically obtained after the interaction, detecting whether a dialog is successful or not with the definition of a set of stop conditions.

3. Our proposal to introduce statistical methodologies in commercial dialog systems

As previously discussed, in commercial settings application developers, together with VUI designers, typically hand-craft dialog management strategies using rules and heuristics. As it is extremely challenging to anticipate every possible user input, hand-crafting dialog management strategies is an error-prone process that needs to be iteratively refined and tuned, which requires much time and effort. The statistical approach provides application developers a tool for automating the design of dialog management strategies by learning these strategies from feedback data.

On the other hand, backend integration is also a large part of the engineering of a SDS, as most applications need to interact with external content management systems and tools, such as databases, customer relationship managers (CRMs), computer telephony integration (CTI) with call-center agents, diagnostic tools, Internet repositories, etc. The use of widespread interfaces for these external interactions can leverage the engineering task and enhance user experience (e.g., reducing delays in the response of the backend).

In the following subsection we propose a methodology for statistical modeling dialog, which has the advantages of machine learning approaches and, at the same time, makes it possible to use the current technologies and available standards because it generates VoiceXML code that is compliant with commercial platforms. The proposed methodology is based on a classification process that selects the best system answers based on observed user-system interactions. Then, the selected dialog state is generated using

VoiceXML so that it can be correctly interpreted in the industry platforms.

Additionally, as the success of statistical approaches depends on the quality of the data used to develop the dialog model, considerable effort is necessary to acquire and label a corpus with the data necessary to train a good model. In addition, the usability of many commercial spoken dialog systems is still limited, as developers frequently do not have the time to perform user tests during the development cycle. In order to avoid these negative effects, we also propose an approach to develop a user simulator and automatically acquire a dialog corpus. This way, the user and dialog models are iteratively built from the interaction of both modules. Following our approach, a labeled corpus for the specific task is automatically obtained after the interaction, detecting whether a dialog is successful or not with the definition of a set of stop conditions. It only requires the definition of the semantics for the task for both the user simulator and the dialog manager, thus a previous dialog corpus is not necessary.

3.1. Our approach for statistical dialog management

A conventional dialog manager maintains a state n such as a form or frame and relies on two functions for control, G and F . For a given dialog state n , $G(n) = a$ decides which system action to output, and then after observation o has been received, $F(n, o) = n_0$ decides how to update the dialog state n to yield n_0 . This process repeats until the dialog ends. The important point is that G and F are written by hand, for example by means of a language such as VoiceXML.

In a statistical approach, the conventional dialog manager is extended in three respects: firstly, its action selection function $G(n) = a$ is changed to output a set of one or more (M) allowable actions given a dialog state n , $G(n) = \{a_1, a_2, \dots, a_M\}$. Next, its transition function $F(n, o) = n_0$ is extended to allow for different transitions depending on which of these action was taken, $F(n, a, o) = n_0$. A (human) dialog designer still designs the contents of the state n and writes the functions G and F .

As stated in the introduction, our approach to integrate statistical methodologies in commercial applications is based on the automatic learning of the dialog strategy by means of a labeled corpus for the system's task. In most dialog systems, the dialog manager makes decisions based only on the information provided by the user in the previous turns and its own dialog model. For example, this is the case with most

dialog systems for slot-filling tasks. The methodology that we propose for the selection of the next system response for this kind of task is detailed in [27]. We represent the dialogs as a sequence of pairs (A_i, U_i) , where A_i is the output of the dialog system (the system answer) at time i , expressed in terms of dialog acts; and U_i is the semantic representation of the user turn (the result of the understanding process of the user input) at time i , expressed in terms of frames. This way, each dialog is represented by:

$$(A_1, U_1), \dots, (A_i, U_i), \dots, (A_n, U_n)$$

where A_1 is the greeting turn of the system, and U_n is the last user turn. From now on, we refer to a pair (A_i, U_i) as S_i , the state of the dialog sequence at time i .

In this framework, we consider that at time i , the objective of the dialog manager is to find the best system answer A_i . This selection is a local process for each time i which takes into account the previous history of the dialog, i.e., the sequence of states of the dialog preceding time i :

$$\hat{A}_i = \operatorname{argmax}_{A_i \in \mathcal{A}} P(A_i | S_1, \dots, S_{i-1}) \quad (1)$$

where the set \mathcal{A} contains all the possible system answers.

The main problem to resolve this equation is regarding the number of possible sequences of states, which is usually very large. To solve the problem, we define a data structure in order to establish a partition in this space, i.e., in the history of the dialog preceding time i). This data structure, which we call *Dialog Register* (DR), contains the information provided by the user throughout the previous history of the dialog. After applying the above considerations and establishing the equivalence relation in the histories of dialogs, the selection of the best A_i is given by:

$$\hat{A}_i = \operatorname{argmax}_{A_i \in \mathcal{A}} P(A_i | DR_{i-1}, S_{i-1}). \quad (2)$$

Each user turn supplies the system with information about the task; i.e., the user asks for a specific concept and/or provides specific values for certain attributes. However, a user turn can also provide other kinds of information, such as task-independent information (for instance, *Affirmation*, *Negation*, and *Not-Understood* dialog acts). This kind of information implies some decisions which are different from simply updating the

DR_{i-1} . Hence, for the selection of the best system response A_i , we take into account the DR that results from turn 1 to turn $i-1$, and we explicitly consider the last state S_{i-1} .

We propose to solve Eq. (2) by means of a classification process. This way, every dialog situation (i.e., each possible sequence of dialog acts) is classified taking into a set of classes \mathcal{C} , which groups together all the sequences that provide the same set of system actions (answers). The objective of the dialog manager at each moment is to select a class of this set $c \in \mathcal{C}$, thus the answer of the system at that moment is the answer associated with the selected class. To implement this procedure we use a multilayer perceptron (MLP) [53] where the input layer receives the input pair (DR_{i-1}, S_{i-1}) corresponding to the dialog register and the state. The values of the output layer can be seen as an approximation of the a posteriori probability of the input belonging to the associated class $c \in \mathcal{C}$.

As stated before, the DR contains information about concepts and attributes provided by the user throughout the previous history of the dialog. For the dialog manager to determine the next answer, we have assumed that the exact values of the attributes are not significant. They are important for accessing the databases and for constructing the output sentences of the system. However, the only information necessary to predict the next action by the system is the presence or absence of concepts and attributes. Therefore, the information we used from the DR is a codification of this data in terms of three values, $\{0, 1, 2\}$, for each field in the DR according to the following criteria: (0) The concept is unknown or the value of the attribute is not given; (1) the concept or attribute is known with a confidence score that is higher than a given threshold; (2) the concept or attribute has a confidence score that is lower than the given threshold.

3.2. User simulation to learn the dialog model

We propose to use a dialog simulation technique in order to automatically acquire the corpus that is required to learn the dialog model and train the neural network for the statistical dialog manager. Our approach for acquiring a dialog corpus is based on the interaction of a user simulator and a dialog manager simulator [26]. Both modules use a random selection of one of the possible answers defined for the semantics of the task (user and system dialog acts). At the beginning of the simulation, all system answers are defined as equiprobable. When a successful dialog is

simulated, the probabilities of the answers selected by the dialog manager during that dialog are incremented before starting a new simulation.

This technique simulates the user intention level, i.e., the simulator provides concepts and attributes that represent the intention of the user utterance. An error simulation module has been implemented to include semantic errors in the generation of dialogs. Once the user simulator has selected the information to be provided to the user, the error simulator modifies the frames created and provides a confidence score for each concept and attribute in the semantic representation of the user turn.

The model employed for introducing errors and confidence scores is inspired in the one presented in [57]. Both processes are carried out separately following the noisy communication channel metaphor by means of a generative probabilistic model $P(c, U_a | \tilde{U}_a)$, where U_a is the true incoming user dialog act, \tilde{U}_a is the recognized hypothesis, and c is the confidence score associated with this hypothesis.

On the one hand, the probability $P(\tilde{U}_a | U_a)$ is obtained by Maximum-Likelihood using the initial labeled corpus acquired with real users. To compute it, we consider the recognized sequence of words w_U and the actual sequence uttered by the user \tilde{w}_U . This probability is decomposed into a component that generates the word-level utterance that corresponds to a given user dialog act, a model that simulates ASR confusions (learned from the reference transcriptions and the ASR outputs), and a component that models the semantic decoding process.

$$\begin{aligned} P(\tilde{U}_a | U_a) \\ = \sum_{\tilde{w}_U} P(U_a | \tilde{w}_U) \sum_{w_U} P(\tilde{w}_U | w_U) P(w_U | U_a). \end{aligned}$$

On the other hand, the generation of confidence scores is carried out by approximating $P(c | \tilde{U}_a, U_a)$ assuming that there are two distributions for c . These two distributions are defined manually generating confidence scores for correct and incorrect hypotheses. These definitions are based on a sampling over the distributions found in the training data corresponding to our initial corpus.

$$P(c | U_a, \tilde{U}_a) = \begin{cases} P_{corr}(c) & \text{if } \tilde{U}_a = U_a, \\ P_{incorr}(c) & \text{if } \tilde{U}_a \neq U_a. \end{cases}$$

A maximum number of turns per dialog is considered for acquiring a corpus using our user simulator,

taking into account the requirements of the task for real users. A user request for closing the dialog is selected once the system has provided the information defined in the objective(s) of the dialog. The dialogs that fulfill this condition before the maximum number of turns are considered successful. The dialog manager considers that the dialog is unsuccessful and decides to abort it when the following conditions take place: i) The dialog exceeds the maximum number of user turns; ii) The answer selected by the dialog manager corresponds to a query not made by the user simulator; iii) The database query module generates an error because the user simulator has not provided the mandatory data to carry out the query; iv) The answer generator generates an error because the selected answer involves the use of a data not provided by the user simulator.

3.3. Implementation by means of the VoiceXML standard

As discussed above, to improve the current technology we propose to merge statistical approaches with VoiceXML. Our goal is to combine the flexibility of statistical dialog management with the facilities that VoiceXML offers, which would help to introduce statistical approaches for the development of commercial (and not strictly academic) dialog systems. To this end, our proposal employs the described statistical dialog management technique to decide the next system prompt. In addition, the system prompts and the grammars for ASR are implemented in VoiceXML compliant formats, for example, JSGF or SRGS.

In contrast to other hybrid approaches, our main aim is not to incorporate knowledge about best strategies in statistical dialog management, but rather to take advantage of an implementation language which has been traditionally used to build rule-based systems (such as VoiceXML), for the development of statistical dialog strategies. Expert knowledge about deployment of VoiceXML applications, development environments and tools can still be exploited using our technique. The only change is in the transition between states, which is carried out on a data-driven basis.

Figure 3 shows the architecture designed for the integration of the statistical methodology for dialog management with the functionalities provided by VoiceXML. As can be observed, a VoiceXML-compliant platform (such as Voxeo Evolution⁶) is used

⁶<http://evolution.voxeo.com/>

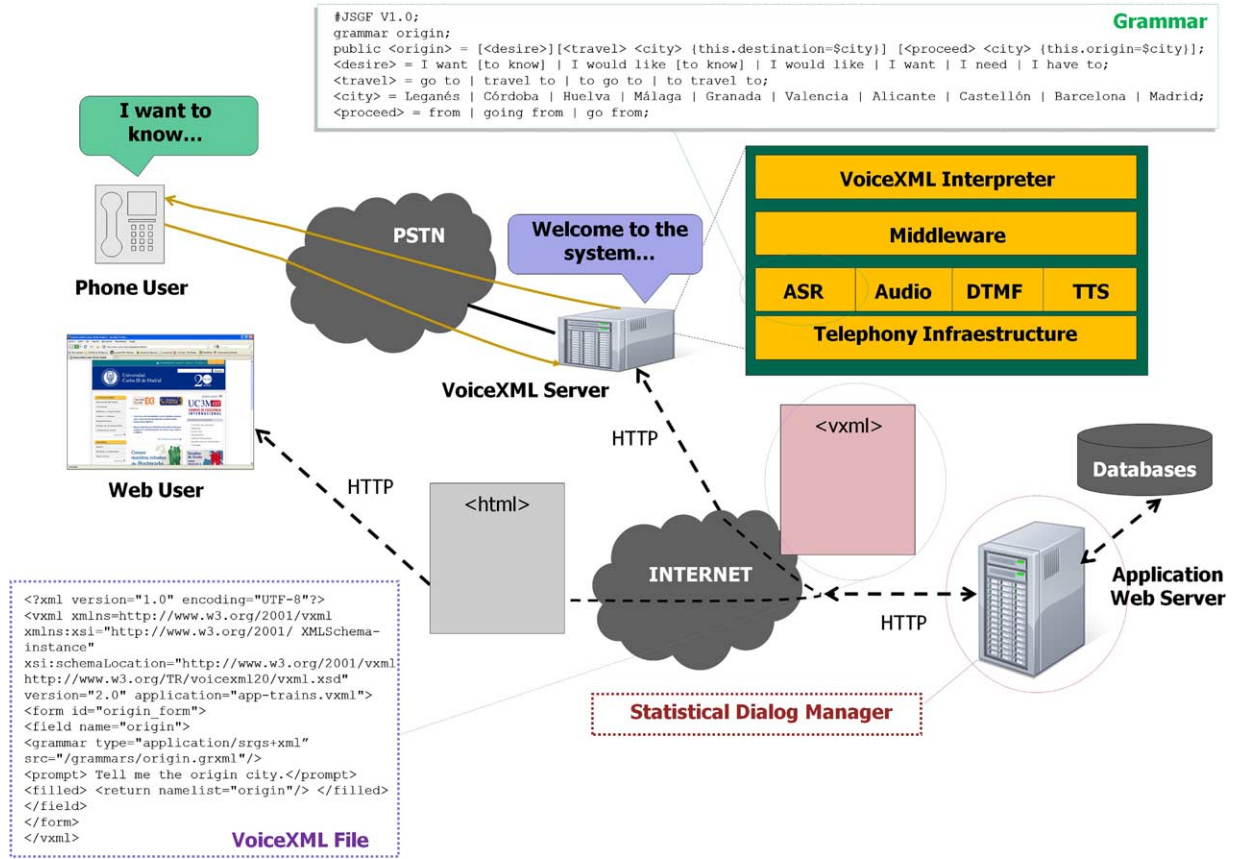


Fig. 3. Architecture for the integration of the proposed statistical methodology for dialog management with VoiceXML.

to create Interactive Voice Response applications and the provision of telephone access. Static VoiceXML files and grammars can be stored in the voice server. We propose to simplify these files by generating a VoiceXML file for each specific system prompt, as can be observed in the bottom left corner of the figure. Each file contains a reference to a grammar that defines the valid user's inputs for the corresponding system prompt, as observed at the right top corner of the figure.

The dialog system selects the next system prompt (i.e. VoiceXML file) by verifying the probabilities assigned by the statistical dialog manager to each system prompt given the current state of the dialog. This module is stored in an external web server and is implemented using a data structure to store the information that is provided by the user in each dialog turn (i.e., the *Dialog Register*) and a call to the trained neural network using this information and the last system prompt, i.e., the codification of the pair (DR_{i-1}, S_{i-1}) , which represents the current state of

the dialog. The result generated by the statistical dialog manager informs the IVR platform about the most probable system prompt to be selected for the current dialog state. The platform just selects the corresponding VoiceXML file and reproduces it to the user.

4. Development of a practical dialog system

To test our proposal, we have used the definitions of the Universidad Al Habla (UAH – University on the Line) dialog system. UAH is a spoken dialog system developed to provide spoken access to academic information about the Department of Languages and Computer Systems in the University of Granada, Spain [8,10]. The system is comprised of the five typical modules of current spoken dialog systems described in Section 2, concerned with automatic speech recognition, dialog management, database access, data storage, and oral response generation. An additional module was developed to automatically create dynamic ASR grammars [9].

Table 1
Information provided by the UAH system

Category	Information provided by the user (names and examples)		Information provided by the system
Subject	<i>Name</i>	Compilers	Degree, professors, responsible professor, semester, credits, web page
	<i>Degree</i> , in which it is taught in case that there are several subjects with the same name	Computer science	
	<i>Group name</i> and optionally <i>type</i> , in case he asks for information about a specific group	A Theory A	Timetable, professor
Professors	Any combination of <i>name</i> and <i>surnames</i>	Zoraida Zoraida Callejas Ms. Callejas	Office location, contact information (phone, fax, email), groups and subjects, doctoral courses
	Optionally <i>semester</i> , in case he asks for the tutoring hours	First semester Second semester	Tutoring hours
Doctoral studies	Name of a doctoral program	Software development	Department, responsible
	Name of a course if he asks for information about a specific course	Object-oriented programming	Type, credits
Registration	Name of the deadline	Provisional registration confirmation	Initial time, final time, description

The information that the system provides can be classified in four main groups: subjects, professors, doctoral studies and registration, as shown in Table 1. As can be observed, the system must ask the user for different pieces of information before producing a response. The way in which the user is queried for this information follows in most cases a system-directed initiative.

A set of 500 dialogs was acquired with an initial version of the UAH system by means of its interaction with students and professors of the University of Granada. A rule-based strategy for dialog management was designed for this initial system based on VoiceXML files [8]. This system has been used as a baseline to compare with the system developed following the approach presented in this paper. The acquisition process using this initial system resulted in a spontaneous Spanish speech dialog corpus with 125 different speakers, with small dialectal variants. The total number of user turns was 2110. The recorded material has a duration of 9.5 hours. The characteristics of the acquired corpus are summarized in Table 2. Figure 4 shows an example of a dialog extracted from this corpus.

In order to apply the proposed statistical dialog management methodology, we defined a semantic representation for the UAH task. As in many other dialog systems, the semantic representation of the user and system turns is represented in terms of frames [41]. A frame is a structure for representing semantic knowledge relate to a concept or situation. Each concept in a domain has usually associated a group of attributes (slots) and values [20]. This way, using this representation, one or more concepts usually represent the intention of the utterance, and a sequence of attribute-value pairs contains the information about the values given by the user. Therefore, the NLU module takes the sentence supplied by the recognition process as input and generates one or more frames as output. The confidence scores provided by the NLU module to determine the reliability of the data provided by the user are shown between brackets. In the case of the system turns, the NLG module of this systems translates the semantic representations of the system dialog acts, generated by the dialog manager, to sentences in Spanish.

In the case of user turns, we defined four concepts related to the different queries that the user can

Table 2

Main characteristics of the corpus acquired with the initial version of the UAH system

Number of users	125
Number of user turns	2110
Average number of user turns per dialog	4.2
Average number of words per user turn	6.7
Duration of the recording (hours)	9.5

S1: Welcome to the UAH system. You can consult information about subjects, professors, doctoral studies and registrations.

U1: I want to know information about subjects in Computer Science

S2: Which subject do you want to consult?

U2: Language Processors I

S3: Here you can consult the information of the subject. Anything else?

U3: Can you inform me about the schedules of Group D?

S4: I give the information for the Group D of Language Processors I: Monday 9:00-11:00, Wednesday 11:00-13:00. Anything else?

U4: Can you tell me the name of the professor?

S5: I give the information for the Group D of Language Processors I. The name of the professor is David Griol. Anything else?

U5: No, thank you.

S6: Thank you for using the UAH system.

Fig. 4. An example of dialog for the UAH task (translated from Spanish).

perform to the system (*Subject, Professors, Doctoral studies, Registration*), three task-independent concepts (*Affirmation, Negation, and Not-Understood*), and eight attributes (*Subject-Name, Degree, Group-Name, Subject-Type, Professor-Name, Program-Name, Semester and Deadline*). An example of the semantic interpretation of a user's sentence is shown below:

User Turn:

I want to know information about the subject Language Processors I of Computer Science.

Semantic Representation:

(*Subject*)

Subject-Name: Language Processors I

Degree: Computer Science

The labeling of the system turns is similar to the labeling defined for the user turns. A total of 30 task-dependent concepts was defined:

- Task-independent concepts (*Opening, New-Query, Affirmation, Negation, Not-Understood and Closing*).
- Concepts used to inform the user about the result of a specific query (*Subject, Professors, Doctoral-Studies and Registration*).
- Concepts defined to require the user the attributes that are necessary for a specific query (*Subject-Name, Degree, Group-Name, Subject-Type, Professor-Name, Program-Name, Semester and Deadline*).
- Concepts used for the confirmation of concepts (*Confirmation-Subject, Confirmation-Lecturers, Confirmation-DoctoralStudies, Confirmation-Registration*) and attributes (*Confirmation-Subject-Name, Confirmation-Degree, Confirmation-Group-Name, Confirmation-SubjectType, Confirmation-ProfessorName, Confirmation-ProgramName, Confirmation-Semester and Confirmation-Deadline*).

An example of the semantic interpretation of a system prompt is shown below:

System Turn:

Do you want to know the name of the professor of the Group 1 of Language Processors I?

Semantic Representation:

(*Professors*)

Group-Name: 1

Subject-Name: Language Processors I

The *DR* defined for the UAH task is a sequence of 12 fields, corresponding to the four concepts (*Subject, Professors, Doctoral-Studies and Registration*) and eight attributes (*Subject-Name, Degree, Group-Name, Subject-Type, Professor-Name, Program-Name, Semester and Deadline*) defined for the task.

Using the codification previously described for the information in the *DR*, every dialog begins with a dialog register in which every value is equal to "0" and the greeting turn of the system. We represent this initial *DR* as 0000 – 00000000, where the first part represents the possible concepts in the task and the second part denotes the possible attributes, as it is showed following.

.....

S₁: Welcome to the UAH system. You can consult information about subjects, professors, doctoral studies and registrations.

A₁: (*Opening*)

*DR*₀: 0000-00000000

.....

Each time the user provides information, this is used to update the previous *DR* and to obtain the new one. For instance, given a user turn with a query about subjects providing the name of the subject and the degree, the new dialog register could be as follows. Confidence scores provided by the NLU module to determine the reliability are shown between brackets.

.....

*U*₁: I want to know information about the subject *Language Processors I* in the *Computer Science* degree.

Task-Dependent Information: (*Subject*) [0.81]

Subject-Name: *Language Processors I* [0.36]

Degree: *Computer Science* [0.93]

Task-Independent Information: None

*DR*₁: 1000-21000000

.....

In this case, the confidence score assigned to the attribute *Subject-Name* is very low. Then, a “2” value is added in the corresponding position of the *DR*₁. The concept (*Subject*) and the attribute *Degree* are recognized with a high confidence score, adding a “1” value in the corresponding positions of the *DR*₁. Then, the input of the MLP is generated using *DR*₁, the codification of the labeling of the last system turn (*A*₁), and the task-independent information provided in the last user turn (none in this case). The output selected for the MLP would consist in this case of a confirmation of the name of the subject. This process is repeated to predict the next system response afterwards each user turn.

5. Evaluation methodology and measures

It is very difficult to define new procedures and measures unanimously accepted by the scientific community for the evaluation of voice-based systems. In fact, this field can be considered to be in an initial phase of development. In [63] and [59], a set of statistical measures to evaluate the quality of the simulated corpus is proposed. Three dimensions are defined: high-level features (dialog and turn lengths), dialog style (speech-act frequency; proportion of goal-directed actions, grounding, formalities, and unrecognized actions; proportion of information provided, re-provided, requested and rerequested), and dialog efficiency (goal completion rates and times). The simulation presented in [2,55,56] is evaluated by testing the similarity between real and simulated data by means of statistical measures (dialog length, task completion rate and dialog performance).

We have adapted the previously described measures considering the information that is available in the definition of the dialog system. Our proposed measures can be classified into three groups: task success/efficiency measures, high-level dialog features, and dialog style/cooperativeness measures.

- Task success/efficiency measures: These measures study the goal achievement rates and goal completion times for the services provided by the system.
- High-level dialog features: These features evaluate how long the dialogs last, how much information is transmitted in individual turns, and how active the dialog participants are.
- Dialog style/cooperativeness measures: These measures analyze the frequency of different speech acts and study what proportion of actions is goal-directed, what part is taken up by dialog formalities, etc.

By means of task success/efficiency measures in our evaluation, we investigate the success rate and efficiency of the services provided by the system. We are particularly interested in goal achievement rates and goal completion times. Only the dialogs were considered successful if they fulfill the complete list of objectives that has been previously defined for it.

Six high-level dialog features have been defined for the evaluation of the dialogs: the average number of turns per dialog, the percentage of different dialogs without considering the attribute values, the number of repetitions of the most seen dialog, the number of turns of the most seen dialog, the number of turns of the shortest dialog, and the number of turns of the longest dialog. Using these measures, we tried to evaluate the success of the simulated dialogs as well as its efficiency and variability with regard to the different services.

For dialog style features, we define and count a set of system/user dialog acts. On the system side, we have measured the confirmation of concepts and attributes, questions to require information, and system answers generated after a database query. On the user side, we have measured the percentage of turns in which the user carries out a request to the system, provides information, confirms a concept or attribute, the Yes/No answers, and other answers not included in the previous categories.

The previous measures evaluate the overall quality of the acquired dialogs and provided services as a whole. In addition, we have carried out a specific

<pre> <?xml version="1.0" encoding="UTF-8"?> <vxml xmlns="http://www.w3.org/2001/vxml" xmlns:xsi="http://www.w3.org/2001/ XMLSchema-instance" xsi:schemaLocation="http://www.w3.org/2001/vxml http://www.w3.org/TR/voicexml20/vxml.xsd" version="2.0" application="app-UAH.vxml"> <form id="subject_form"> <field name="subject_name"> <grammar type="application/srgs+xml" src="/grammars/subjects.grxml"/> <prompt>Tell me the name of the subject.</prompt> <filled> <return namelist="subject_name"/> </filled> </field> </form> </vxml> </pre>	<pre> #JSGF V1.0; grammar subjects; public <subject_name> = [<desire>] [<information>] {this.subjects=\$subject_name} [<connector> {this.subjects=\$degree}] <desire> = I want [to know] I would like [to know] I would like I want I need To know about; <information> = information about the subject information about the subject; <subject_name> = Language Processors Compilers Computers Basics Physics for Computer Science Computer Technology Statistics Databases Operating Systems I Operating Systems II Formal Languages and Automata Theory Requirements Engineering Computer Networks Artificial Intelligence Logic Design ... ; <connector> = of the of ... ; <degree> = Computer Science Biotechnology Telecommunication ... ; </pre>
---	---

Fig. 5. VoiceXML document to prompt the user for the name of a subject (left) and grammar to recognize this specific information (right).

evaluation of the operation of our dialog management methodology. From our previous work on statistical dialog management [27], we propose four measures to evaluate the performance of the dialog manager.

The first measure, which we call *Unseen Situations*, makes reference to the percentage of dialog situations that are present in the test partition but are not present in the corpus used for learning the dialog manager. The other three measures are calculated by comparing the answer automatically generated by the dialog manager for each input in the test partition with regard to the reference answer annotated in the corpus. This way, the evaluation is carried out turn by turn. These three measures are:

- *Exact Responses*: the percentage of responses provided by the dialog manager that are equal to reference answer in the corresponding turn of the training corpus;
- *Correct Responses*: the percentage of answers provided by the dialog manager that are coherent with the current state of the dialog although they are not the same that the reference answer;
- *Error Responses*: the percentage of answers provided by the dialog manager that would cause the failure of the dialog.

6. Evaluation of the developed dialog system

A total of 500 successful dialogs were generated using the proposed user simulation technique and two types of scenarios: S1 and S2. Scenarios of type S1

define one objective for the dialog (e.g., to know the name of the professor of a specific subject and group), whereas those of type S2 define two (e.g., to consult the information about a subject and carry out a query about a specific deadline). The 47 different prompts the system can provide where then automatically generated by means of a specific VoiceXML file using the proposed technique. For example, Fig. 5 shows the VoiceXML document to prompt the user for the origin city, and the grammar used for ASR.

Figure 6 shows an example of a simulated dialog from the acquired corpus corresponding to a scenario with two objectives (to know timetables of a given professor and subject, and information about the registration in its specific degree). The confidence scores generated by the error simulator are shown between brackets. Sentences in natural language, that are equivalent to the labeled system and user turns, have been incorporated to clarify the explanation.

After the welcome prompt of the system, the user simulator asks for information about professors. The error simulator assigns a low confidence to what it has understood (0.1), so its selects to confirm this query in S2. Then, the user simulator confirms that they had asked about professors and a high confidence value is assigned to this response in turn U2. Next, the system asks for the name of the professor (S3), which is provided by the user simulator in turn U3. Again, a low confidence value is assigned to this information, and the name of the professor is confirmed in turn S4. Once the user simulator has confirmed that the provided value for the name of the subject is valid

S1: Opening
Welcome to the UAH system. You can consult information about subjects, professors, doctoral studies and registrations.

U1: Professors [0.1]
I want to know information about professors.

S2: (Confirmation:Professors)
Do you want to know information about professors?

U2: Affirmation [0.9]
Yes

S3: (Question:Professor_Name)
Tell me the name of the professor.

U3: Professor_Name: Zoraida Callejas [0.1]
Zoraida Callejas

S4: (Confirmation:Professor_Name)
Do you want to know the information about Zoraida Callejas?

U4: Affirmation [0.9]
Yes

S5: (Answer:Professors) (New-Query)
Zoraida Callejas teaches the subject Object Oriented Programming and Design. Her office number is 3.23 Anything else?

U5: (Question:Registration) [0.9]
Degree: Computer Science[0.9]
The registration information in Computer Science

S6: (Answer:Registration) (New-Query)
You have until Dec 8th 2011 to register. Anything else?

U6: Negation

S7: (Closing:Nil:Nil)
Thank you for using the UAH system.

Fig. 6. An example of a dialog acquired by means of the simulation technique.

(turn U4), the system provides the specific information in turn S5. In the following turn the user simulator asks for registration deadlines for a specific degree (turn U5). Given that the error simulator assigns a high confidence value to this information, the system directly provides an answer to this query in turn S6. Once the user simulator indicates that no additional information is required to fulfill the objectives of this dialog, the system generates a closing prompt (S7).

In order to employ this corpus to successfully use neural networks as classifiers for the statistical dialog manager, a number of considerations had to be taken into account, such as the network topology, the training

algorithm, and the selection of the parameters of the algorithm. To train and evaluate the neural networks, we used the *April* toolkit, developed by the Technical University of Valencia [18]. Using *April*, topology and algorithm parameters (i.e. learning rate and momentum) are estimated with an exhaustive search, using as stop criteria the mean squared error (MSE) obtained in each epoch for the validation set. The gradient is computed in incremental mode, this way the weights are updated after each input, also a momentum is added to the backpropagation so that the networks can overcome local minimums. Different experiments were conducted using different network topologies of increasing number of weights: a hidden layer with 2 units, two hidden layers of 2 units each, two hidden layers of 4 and 2 units, a hidden layer with 4 units, etc. Several learning algorithms were also tested: the incremental version of the backpropagation algorithm (with and without momentum term) and the quickprop algorithm. The influence of their parameters such as learning rate or momentum term was also studied.

We firstly tested the influence of the topology of the MLP, by training different MLPs of increasing number of weights using the standard backpropagation algorithm (with a sigmoid activation function and a learning rate equal to 0.2), and selecting the best topology according to the mean square error (MSE) of the validation data. The minimum MSE of the validation data was achieved using an MLP of one hidden layer of 32 units. We followed our experimentation with MLPs of this topology, training MLPs with several algorithms: the incremental version of the backpropagation algorithm (with and without momentum term) and the quickprop algorithm. The best result on the validation data was obtained using the MLP trained with the standard backpropagation algorithm and a value of LR equal to 0.3.

6.1. Results of the high level and dialog style features

To compare the corpora acquired with the initial rule-based DM developed for the UAH system and the system including the statistical DM, we computed the mean value for each corpus with respect to each of the evaluation measures shown in the previous section. Two-tailed t-tests have been used to compare the means across the two corpora as described in [2]. All differences reported as statistically significant have p-values less than 0.05 after Bonferroni corrections.

As stated in Section 5, the first group of experiments covers the following statistical properties: i) Dialog

Table 3
Results of the high-level dialog features defined for the comparison of both systems

	Initial System	Statistical System
Average number of user turns per dialog	4.22	3.05
Percentage of different dialogs	84.45%	78.82%
Number of repetitions of the most seen dialog	5	7
Number of turns of the most seen dialog	4	2
Number of turns of the shortest dialog	2	2
Number of turns of the longest dialog	14	12

Table 4
Percentages of the different types of system dialog acts in both systems

	Initial System	Statistical System
Confirmation of concepts and attributes	13.51%	11.23%
Questions to require information	18.44%	16.57%
Answers generated after a database query	68.05%	72.20%

length, measured in the average number of turns per dialog, number of turns of the shortest dialog, number of turns of the longest dialog, and number of turns of the most seen dialog; ii) Different dialogs in each corpus, measured in the percentage of different dialogs and the number of repetitions of the most seen dialog; iii) Turn length, measured in the number of actions per turn; iv) Participant activity as a ratio of system and user actions per dialog.

Table 3 shows the results of the comparison of the high-level dialog features. One of the most significant differences is the average number of user turns. In the two types of scenarios, the dialogs acquired using the simulation technique are shorter than those acquired with real users. This fact can be explained due there are a set of dialogs acquired with real users in which the user asked for additional information not included in the definition of the corresponding scenario once its objectives were achieved.

The number of different dialogs is also lower using the statistical system, due to the reduction in the number of turns, as it can be observed in the number of repetitions of the most seen dialog. This is because users have more variability in order to provide the different information that is needed to access the different queries in the initial system.

The mean values of the turn length and dialog length for the real and statistical corpus are almost the same. The dialogs acquired with the automatic generation technique are statistically shorter, as they provide 1.24 actions per user turn instead of the 1.08 actions provided by the real users. The shape of the distributions showed that the real dialogs have the largest standard

Table 5
Percentages of the different types of user dialog acts in both systems

	Initial System	Statistical System
Request to the system	31.74%	33.43%
Provide information	21.72%	19.98%
Confirmation	10.81%	9.34%
Yes/No answers	33.47%	35.77%
Other answers	2.26%	1.48%

deviation given that the task length of these dialogs is more disperse. The dialogs acquired using the automatic generation technique have the minor deviation since the successfully simulated dialogs are usually those that use the minor number of turns to achieve the objective(s) predefined.

Regarding the dialog participant activity, the real and statistical corpora have almost exact values for the ratio of user versus system actions. The proportion of system actions in the corpus acquired using the automatic dialog generation technique is only slightly higher (49.8% for the initial corpus and 50.3% for the simulated corpus).

Tables 4 and 5 respectively show the frequency of the most dominant user and system dialog acts in the initial and the statistical systems. In both cases, it can be observed that there are slight differences in the dialog acts distribution. Table 4 compares the percentage of most dominant system dialog acts (classified into confirmations of concepts and attributes, questions to require data from the user, and answers obtained after a query to the database) automatically provided by the dialog manager after this evaluation. There is a higher percentage of confirmations and questions in the cor-

Table 6

Percentages of goal directed and grounding actions in both systems

	Initial System	Statistical System
Goal directed actions	62.24%	71.56%
Grounding actions	36.48%	26.98%
Rest of actions	1.28%	1.46%

pus acquired with the initial system because there is a set of the dialogs in which the users asked for additional information although it was not stated in the used scenario, which also implies this difference in the percentage of the most dominant user dialog acts in the simulated corpus. There is also a reduction in the system requests in the statistical system. This explains a higher proportion of the inform and confirmation system actions in this system.

Regarding the percentage of most dominant user dialog acts (classified into requests to the system, provide information, confirmations, yes/no answers, and other answers), from the results obtained in Table 5, it can be observed that users need to confirm and provide less information using the statistical system. This explains the higher proportion for the rest of user actions in this system. It can also be observed a higher proportion of yes/no actions for the dialogs acquired with the statistical system. These actions are mainly used to confirm that the specific information has been correctly provided using this system.

Finally, we grouped all user and system actions into three categories: “goal directed” (actions to provide or request information), “grounding” (confirmations and negations), and “rest”. Table 6 shows a comparison between these categories. As can be observed, the dialogs provided by the statistical system have a better quality, as the proportion of goal-directed actions is higher.

6.2. Evaluation of the dialog management methodology

Two dialog managers were developed using respectively each one of the two corpora of 500 dialogs acquired using respectively the initial and the statistical UAH systems. A 5-fold cross-validation process was used to carry out the evaluation of both managers. Each one of the acquired corpus was randomly split into five subsets 20% of the corpus). Our experiment consisted of five trials. Each trial used a different subset taken from the five subsets as the test set, and the remaining 80% of the corpus was used as the training set. A validation subset (20%) was extracted from each training set.

Table 7

Results of the evaluation of the dialog management methodology for the initial and statistical UAH systems

	Initial System	Statistical System
Unseen situations	20.98%	18.24%
Exact Responses	85.76%	90.89%
Correct Responses	94.33%	97.43%
Erroneous Responses	3.55%	2.57%

Table 7 shows the results of the evaluation of the different measures proposed. From the results, it can be seen that the number of unseen situations (not present in the training corpus) is reduced in the statistical system due to by using the statistical dialog system, the variability of the different dialogs is reduced (given that the number of turns is also reduced). This is the main cause of obtaining better results for the rest of measures in the evaluation of the dialog manager developed for the statistical system.

The results of the *Exact Responses* and *Correct Responses* measures show the satisfactory operation of the developed dialog manager for both systems. The codification developed to represent the state of the dialog and the good operation of the MLP classifier make it possible for the answer generated by the manager to agree with the reference answer for the corresponding turn by a percentage of 85.76% and 90.89% respectively for the initial and statistical systems.

Finally, the number of answers generated by the MLP that can cause the failure of the system is only a 2.57% for the statistical system. An answer that is coherent with the current state of the dialog is generated in 97.43% of cases for this system. These last two results also demonstrate the correct operation of the classification methodology developed to deal with the uncertainty of the complete set of possible situations during a dialog.

6.3. Evaluation with real users

Finally, we evaluated the behavior of our two systems with real users using the same set of type S1 and S2 scenarios designed for the user simulation. A total of 150 dialogs were recorded from interactions of six users employing the initial version and the system developed using the approach presented in this paper. An objective and subjective evaluation were carried out. We considered the following measures for the objective evaluation:

- *Successful dialogs*. This is the percentage of successfully completed tasks. In each scenario, the

Table 8
Results of the objective evaluation of the initial and statistical UAH systems with real users

	Successful dialogs	nT	Confirmation rate	ECR	nCE	nNCE
Initial system	85%	11.4	34%	82%	0.84	0.18
Statistical system	94%	8.2	26%	91%	0.88	0.09

- user has to obtain one or several items of information, and the dialog success depends on whether the system provides correct data (according to the aims of the scenario) or incorrect data to the user.
- *Average number of turns per dialog (nT)*.
 - *Confirmation rate*. It was computed as the ratio between the number of explicit confirmations turns (nCT) and the number of turns in the dialog (nCT/nT).
 - *Average number of corrected errors per dialog (nCE)*. This is the average of errors detected and corrected by the dialog manager. We have considered only those errors that modify the values of the attributes and that could cause the failure of the dialog.
 - *Average number of uncorrected errors per dialog (nNCE)*. This is the average of errors not corrected by the dialog manager. Again, only errors that modify the values of the attributes are considered.
 - *Error correction rate (ECR)*. The percentage of corrected errors, computed as $nCE / (nCE + nNCE)$.

Table 8 presents the results of the objective evaluation. These results show that both systems could interact correctly with the users in most cases. However, the statistical system obtained a higher success rate, improving the initial results by 9% absolute. Using the statistical system, the average number of required turns is also reduced from 11.4 to 8.2. These values are slightly higher for both systems regarding the results obtained with the user simulator as in some dialogs the real users provided additional information which was not mandatory for the corresponding scenario or asked for additional information not included in the definition of the scenario once its objectives were achieved.

Table 9 shows a comparison for the three categories defined to classify system actions (“goal directed”, “grounding”, and “rest”). As can be observed, the results obtained in the evaluation with real users are more similar to the ones obtained with the user simulator. This way, dialogs provided by the statistical system has again a better quality, as the proportion of goal-directed actions is higher.

Table 9
Percentages of goal directed and grounding actions in both systems

	Initial System	Statistical System
Goal directed actions	64.17%	73.23%
Grounding actions	34.69%	25.24%
Rest of actions	1.14%	1.53%

The confirmation and error correction rates were also improved by the statistical system, as context information makes possible to require less information to the user, reducing the probability of introducing ASR errors. The main problem detected was that when there was a user input misrecognized with a very high ASR confidence, this erroneous information was forwarded to the dialog manager. However, as the success rate shows, this fact did not have a considerable impact on the system operation.

As stated above, data-driven approaches to dialog optimization require an objective criterion that can be easily measured. However, this type of objective criteria is often not sufficient to completely characterize a system’s behavior with respect to the subjective experience of the user. One of the ultimate goals of designing and building spoken dialog systems is therefore the optimization of the caller experience. This way, we also asked the users to complete a questionnaire to assess their subjective opinion about the system performance. The questionnaire had five questions: i) Q1: *How well did the system understand you?*; ii) Q2: *How well did you understand the system messages?*; iii) Q3: *Was it easy for you to get the requested information?*; iv) Q4: *Was the interaction rate adequate?*; v) Q5: *Was it easy for you to correct the system errors?* The possible answers for each one of the questions were the same: *Never/Not at all, Seldom/In some measure, Sometimes/Acceptably, Usually/Well, and Always/Very Well*. All the answers were assigned a numeric value between one and five (in the same order as they appear in the questionnaire). Table 10 shows the average results of the subjective evaluation using the described questionnaire.

From the results, it can be observed that both systems are considered to correctly understand the different user queries and obtain a similar evaluation regarding the facility of correcting errors introduced by

Table 10

Results of the subjective evaluation of the initial and statistical UAH systems with real users (1 = worst, 5 = best evaluation)

	Q1	Q2	Q3	Q4	Q5
Initial System	4.6	3.6	3.8	3.4	3.2
Statistical System	4.7	4.1	4.4	4.5	3.6

the ASR module. However, the statistical system has a higher evaluation rate regarding the facility of obtaining the data required to fulfill the complete set of objectives of the scenario and the suitability of the interaction rate during the dialog.

7. Conclusions and future work

In order to perceive their environment as intelligent, users must be able to communicate with it using natural and flexible interfaces. A solution that is widely adopted in AmI applications is to use spoken dialog systems, which allow the user to establish an oral conversation with the system. However, current commercial dialog systems are expensive to develop and are usually designed for specific application domains.

In this paper, we have described a technique for developing interactive dialog systems which is comprised of two main proposals: i) an hybrid dialog management approach that combines a well known standard like VoiceXML with statistical estimation of optimized dialog strategies, and ii) a user simulation approach that allows to automatically generate the dialog corpus with which the statistical classifier is trained, thus avoiding the high cost of gathering and labeling a corpus with real users.

The main objective of our work was to reduce the gap between academic and commercial systems by reducing the effort required to define optimal dialog strategies and implement the system. Our proposal combines the benefits of statistical methods for dialog management and VoiceXML. The former provide an efficient means to explore a wider range of dialog strategies, whereas the latter makes it possible to benefit from the advantages of using the different tools and platforms that are already available and simplify system development.

We have applied our technique to develop a dialog system that provides academic information, and automatically create VoiceXML documents to prompt the user for data, as well as the necessary grammars for ASR while following a statistical transition model which optimizes the dialog strategy. The evaluation re-

sults show that the technique can predict coherent system answers in most of the cases, which are also highly rated by real users.

For future work we are interested in applying our proposal to multi-domain tasks in order to measure the capability of our methodology to adapt efficiently to AmI contexts that vary dynamically. We also want to combine our proposal to facilitate the interaction using also additional input and output modalities which are different to speech, both including in the *DR* new features related to these additional modalities and combining our proposal with languages and standards defined for multimodal interaction like XHTML+Voice.

Acknowledgements

Research funded by projects CICYT TIN2011-28620-C02-01, CICYT TEC2011-28626-C02-02, CAM CONTEXTS (S2009/TIC-1485), and DPS2008-07029-C02-02.

References

- [1] K. Acomb, J. Bloom, K. Dayanidhi, P. Hunter, P. Krogh, E. Levin and R. Pieraccini, Technical support dialog systems: issues, problems, and solutions, in: *Proc. NAACL-HLT-Dialog '07 Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, NY, USA, 2007, pp. 25–31.
- [2] H. Ai, A. Raux, D. Bohus, M. Eskenazi and D. Litman, Comparing spoken dialog corpora collected with recruited subjects versus real users, in: *Proc. 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium, 2007, pp. 124–131.
- [3] E. Barnard, A. Halberstadt, C. Kotelly and M. Phillips, A consistent approach to designing spoken-dialog systems, in: *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU'99)*, Keystone, Colorado, USA, 1999, pp. 1173–1176.
- [4] M. Bezold and W. Minker, A framework for adapting interactive systems to user behavior, *Journal of Ambient Intelligence and Smart Environments* 2(4) (2010), 369–387, IOS Press.
- [5] D. Bonino and F. Corno, What would you ask to your home if it were intelligent? Exploring user expectations about next-generation homes, *Journal of Ambient Intelligence and Smart Environments* 3(2) (2011), 111–116, IOS Press.
- [6] J. Bos, E. Klein, O. Lemon and T. Oka, DIPPER: Description and formalisation of an information-state update dialogue system architecture, in: *Proc. 4th SIGdial Workshop on Discourse and Dialogue*, Sapporo, Japan, 2003, pp. 115–124.
- [7] B. Brumitt and J.J. Cadiz, “Let there be light” examining interfaces for homes of the future, in: *Proc. International Conference on Human Computer Interaction (INTERACT'01)*, Amsterdam, The Netherlands, 2001, pp. 375–382.

- [8] Z. Callejas and R. López-Cózar, Implementing modular dialogue systems: A case study, in: *Proc. Applied Spoken Language Interaction in Distributed Environments Conference (ASIDE'05)*, Aalborg, Denmark, 2005.
- [9] Z. Callejas and R. López-Cózar, Automatic creation of ASR grammar rules for unknown vocabulary applications, in: *Proc. 8th International workshop on Electronics, Control, Modelling, Measurement and Signals (ECMS'07)*, Liberec, Czech Republic, 2007, pp. 50–55.
- [10] Z. Callejas and R. López-Cózar, Relations between de-facto criteria in the evaluation of a spoken dialogue system, *Speech Communication* **50**(8–9) (2008), 646–665, Elsevier.
- [11] S.W. Chu, I. O'Neill, P. Hanna and M. McTear, An approach to multistrategy dialogue management, in: *Proc. 9th International Conference on Spoken Language Processing (Interspeech'05-Eurospeech)*, Lisbon, Portugal, 2005, pp. 865–868.
- [12] H.H. Clark and S.E. Brennan, Grounding in communication, in: *The Psychology of Computer Vision*, P.H. Winston, ed., American Psychological Association, 1991, pp. 127–149.
- [13] M. Cohen, J. Giangola, and J. Balough, *Voice User Interface Design*, Addison Wesley, 2004.
- [14] H. Cuayáhuatl, S. Renals, O. Lemon and H. Shimodaira, Human-computer dialogue simulation using hidden markov models, in: *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU'05)*, San Juan, Puerto Rico, 2005, pp. 290–295.
- [15] H. Cuayáhuatl, S. Renals, O. Lemon and H. Shimodaira, Learning multi-goal dialogue strategies using reinforcement learning with reduced state-action spaces, in: *Proc. 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, Pittsburgh, USA, 2006, pp. 469–472.
- [16] W. Eckert, E. Levin and R. Pieraccini, User modeling for spoken dialogue system evaluation, in: *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'97)*, Santa Barbara, USA, 1997, pp. 80–87.
- [17] W. Eckert, E. Levin and R. Pieraccini, Automatic evaluation of spoken dialogue systems, Technical report, TR98.9.1, ATT Labs Research, 1998.
- [18] S. Espana-Boquera, F. Zamora-Martinez, M. Castro-Bleda and J. Gorbe-Moya, Efficient bp algorithms for general feedforward neural networks, *Lecture Notes in Computer Science* **4527** (2007), 327–336, Springer.
- [19] G. Espejo, N. Ábalos, R. López-Cózar, Z. Callejas and D. Griol, System for user location by means of RFID devices for a dialogue system designed to interact in an ambient intelligence environment, in: *Proc. VI Jornadas en Tecnología del Habla and II Iberian SLTech Workshop (FALA 2010)*, Vigo, Spain, 2010, pp. 251–254.
- [20] R.E. Fikes and T. Kehler, The role of frame-based representation in knowledge representation and reasoning, *Communications of the ACM* **28** (1985), 904–920, ACM.
- [21] P. Filipe and N. Mamede, Ambient Intelligence interaction via dialogue systems, in: *Ambient Intelligence*, F.J. Villanueva, ed., In-teh, 2010, pp. 109–124.
- [22] K. Georgila, J. Henderson and O. Lemon, Learning user simulations for information state update dialogue systems, in: *Proc. 9th European Conference on Speech Communication and Technology (Eurospeech'05)*, Lisbon, Portugal, 2005, pp. 893–896.
- [23] K. Georgila, J. Henderson and O. Lemon, User simulation for spoken dialogue systems: Learning and evaluation, in: *Proc. 9th International Conference on Spoken Language Processing (Interspeech/ICSLP)*, Pittsburgh, USA, 2006, pp. 1065–1068.
- [24] J. Glass, J. Polifroni and S. Seneff, Multilingual language generation across multiple domains, in: *Proc. International Conference on Spoken Language Processing (ICSLP'94)*, Yokohama, Japan, 1994, pp. 983–986.
- [25] D. Goddeau, E. Brill, J. Glass, C. Pao, M. Phillips, J. Polifroni, S. Seneff and V. Zue, Galaxy: A human language interface to online travel information, in: *Proc. International Conference on Spoken Language Processing (ICSLP'94)*, Yokohama, Japan, 1994, pp. 707–710.
- [26] D. Griol, Z. Callejas and R. López-Cózar, Acquiring and evaluating a dialog corpus through a dialog simulation technique, in: *Proc. 9th SIGdial Workshop on Discourse and Dialogue*, London, UK, 2009, pp. 326–332.
- [27] D. Griol, L.F. Hurtado, E. Segarra and E. Sanchis, A statistical approach to spoken dialog systems design and evaluation, *Speech Communication* **50**(8–9) (2008), 666–682, Elsevier.
- [28] D. Griol, M.F. McTear, Z. Callejas, R. López-Cózar, N. Ábalos and G. Espejo, A methodology for learning optimal dialog strategies, *Lecture Notes in Computer Science* **6231** (2010), 507–514, Springer.
- [29] D. Griol, G. Riccardi and E. Sanchis, Learning the structure of human-computer and human-human dialogs, in: *Proc. International Conference on Spoken Language Processing (Interspeech'09)*, Brighton, UK, 2009, pp. 2775–2778.
- [30] P. Heeman, Combining reinforcement learning with information-state update rules, in: *Proc. 8th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'07)*, Rochester, New York, USA, 2007, pp. 268–275.
- [31] E. Levin and R. Pieraccini, A stochastic model of human-machine interaction for learning dialog strategies, in: *Proc. 3rd European Conference on Speech Communication and Technology (Eurospeech'97)*, Rhodes, Greece, 1997, pp. 1883–1896.
- [32] E. Levin, R. Pieraccini and W. Eckert, A stochastic model of human-machine interaction for the learning dialog strategies, *IEEE Transactions on Speech and Audio Processing* **8**(1) (2000), 11–23, IEEE.
- [33] R. López-Cózar and M. Araki, *Spoken, Multilingual and Multimodal Dialogue Systems*, John Wiley & Sons, 2005.
- [34] R. López-Cózar and Z. Callejas, Multimodal dialogue for Ambient Intelligence and Smart Environments, in: *Handbook of Ambient Intelligence and Smart Environments*, H. Nakashima, H. Aghajan and J.C. Augusto, eds, Springer, 2010, pp. 559–579.
- [35] R. López-Cózar, Z. Callejas and D. Griol, Using knowledge of misunderstandings to increase the robustness of spoken dialogue systems, *Knowledge-Based Systems* **23**(5) (2010), 471–485, Elsevier.
- [36] P. Lyons, A.T. Cong, H.J. Steinhauer, S. Marsland, J. Dietrich and H.W. Guesgen, Exploring the responsibilities of single-inhabitant Smart Homes with Use Cases, *Journal of Ambient Intelligence and Smart Environments* **2**(3) (2010), 211–232, IOS Press.
- [37] M.F. McTear, *Spoken Dialogue Technology: Towards the Conversational User Interface*, Springer, 2004.
- [38] H.H. Meng, C. Wai and R. Pieraccini, The use of belief networks for mixed-initiative dialog modeling, *IEEE Transactions on Speech and Audio Processing* **11**(6) (2003), 757–773, IEEE.

- [39] D. Milward and M.A. Beveridge, Ontologies and the structure of dialogue, in: *Proc. 8th Workshop on the Semantics and Pragmatics of Dialogue (CATALOG)*, Barcelona, Spain, 2004, pp. 214–216.
- [40] W. Minker, R. López-Cózar and M. McTear, The role of spoken language dialogue interaction in intelligent environments, *Journal of Ambient Intelligence and Smart Environments* **1** (2009), 21–36, IOS Press.
- [41] M. Minsky, A framework for representing knowledge, in: *The Psychology of Computer Vision*, P.H. Winston, ed., McGraw-Hill, 1975, pp. 211–277.
- [42] S. Möller, R. Schleicher, D. Butenkov, K.P. Engelbrecht, F. Gödde, T. Scheffler, R. Roller and N. Reithinger, Usability engineering for spoken dialogue systems via statistical user models, in: *Proc. First International Workshop on Spoken Dialogue Systems Technology (IWSDS'09)*, Irsee, Germany, 2009, pp. 40–47.
- [43] G. Montoro, P.A. Haya and X. Alamán, A dynamic spoken dialogue interface for ambient intelligence interaction, *International Journal of Ambient Computing and Intelligence* **2**(1) (2010), 24–51, IGI.
- [44] T. Paek, Toward evaluation that leads to best practices: Reconciling dialog evaluation in research and industry, in: *Proc. NAACL-HLT-Dialog'07 Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, NY, USA, 2007, pp. 40–47.
- [45] T. Paek and E. Horvitz, Conversation as action under uncertainty, in: *Proc. 16th Conference on Uncertainty in Artificial Intelligence*, San Francisco, USA, 2000, pp. 455–464.
- [46] T. Paek and R. Pieraccini, Automating spoken dialogue management design using machine learning: An industry perspective, *Speech Communication* **50**(8–9) (2008), 716–729, Elsevier.
- [47] R. Pieraccini and J. Huerta, Where do we go from here? Research and commercial spoken dialog systems, in: *Proc. 6th SIGdial Workshop on Discourse and Dialog*, Lisbon, Portugal, 2005, pp. 1–10.
- [48] R. Pieraccini, D. Suendermann, K. Dayanidhi and J. Liscombe, Are we there yet? Research in commercial spoken dialog systems, *Lecture Notes in Computer Science* **5729** (2009), 3–13, Springer.
- [49] O. Pietquin and R. Beaufort, Comparing ASR modeling methods for spoken dialogue simulation and optimal strategy learning, in: *Proc. 9th European Conference on Speech Communication and Technology (Eurospeech'05)*, Lisbon, Portugal, 2005, pp. 861–864.
- [50] O. Pietquin and T. Dutoit, A probabilistic framework for dialog simulation and optimal strategy learning, *IEEE Transactions on Speech and Audio Processing* **14** 2005, 589–599, IEEE.
- [51] J. Polifroni and S. Seneff, GALAXY-II as an Architecture for Spoken Dialogue Evaluation, in: *Proc. Second International Conference on Language Resources and Evaluation (LREC)*, Athens, Greece, 2000, pp. 725–730.
- [52] N. Roy, J. Pineau and S. Thrun, Spoken dialogue management using probabilistic reasoning, in: *Proc. 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, Hong Kong, China, 2000, pp. 93–100.
- [53] D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning internal representations by error propagation, in: *PDP: Computational Models of Cognition and Perception (I)*, D.E. Rumelhart and J.L. McClelland, eds, MIT Press, 1986, pp. 319–362.
- [54] P. Saini, B. de Ruyter, P. Markopoulos and A. van Breemen, Benefits of social intelligence in home dialogue systems, *Lecture Notes in Computer Science* **3585** (2005), 510–521, Springer.
- [55] J. Schatzmann, K. Georgila and S. Young, Quantitative evaluation of user simulation techniques for spoken dialogue systems, in: *Proc. 6th SIGdial Workshop on Discourse and Dialogue*, Lisbon, Portugal, 2005, pp. 45–54.
- [56] J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye and S. Young, Agenda-based user simulation for bootstrapping a POMDP dialogue system, in: *Proc. Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*, Rochester, NY, USA, 2007, pp. 149–152.
- [57] J. Schatzmann, B. Thomson and S. Young, Error simulation for training statistical dialogue systems, in: *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'07)*, Kyoto, Japan, 2007, pp. 526–531.
- [58] J. Schatzmann, B. Thomson and S. Young, Statistical user simulation with a hidden agenda, in: *Proc. 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium, 2007, pp. 273–282.
- [59] J. Schatzmann, K. Weilhammer, M. Stuttle and S. Young, A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies, *Knowledge Engineering Review* **21**(2) (2006), 97–126, Cambridge Journals.
- [60] K. Scheffler and S. Young, Simulation of human-machine dialogues, Technical report, CUED/F-INFENG/TR 355, Cambridge University Engineering Dept., Cambridge, UK, 1999.
- [61] K. Scheffler and S. Young, Probabilistic simulation of human-machine dialogues, in: *Proc. 25th International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00)*, Istanbul, Turkey, 2000, pp. 1217–1220.
- [62] K. Scheffler and S. Young, Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning, in: *Proc. of Human Language Technology Conference (HLT'02)*, San Diego, USA, 2001, pp. 12–18.
- [63] K. Scheffler and S. Young, Corpus-based dialogue simulation for automatic strategy learning and evaluation, in: *Proc. 2nd NAACL Meeting NAACL Workshop on Adaptation in Dialogue Systems*, Pittsburgh, USA, 2001, pp. 64–70.
- [64] S. Singh, M.S. Kearns, D. Litman and M. Walker, Reinforcement learning for spoken dialogue systems, in: *Proc. Neural Information Processing Systems Conference (NIPS'99)*, Denver, USA, 1999, pp. 956–962.
- [65] S. Singh, D. Litman, M. Kearns and M. Walker, Optimizing dialogue management with reinforcement learning: experimenting with the NJFun system, *Journal of Artificial Intelligence*, **16** (2002), 105–133, Elsevier.
- [66] B. Thomson, J. Schatzmann, K. Weilhammer, H. Ye and S. Young, Training a real-world pomdp-based dialog system, in: *Proc. NAACL-HLT-Dialog'07 Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, NY, USA, 2007, pp. 9–16.
- [67] D. Traum and S. Larsson, The information state approach to dialogue management, in: *Current and New Directions in Discourse and Dialogue*, J.C.J. van Kuppevelt and R.W. Smith, eds, Kluwer, 2003, pp. 325–353.
- [68] M. Walker, D. Hindle, J. Fromer, G. Di Fabbrizio and C. Messtel, Evaluating competing agent strategies for a voice email

- agent, in: *Proc. European Conference on Speech Communications and Technology (Eurospeech'97)*, Rhodes, Greece, 1997, pp. 2219–2222.
- [69] C. Wang, J. Glass, H. Meng, J. Polifroni, S. Seneff and V. Zue, Yinhe: A mandarin chinese version of the galaxy system, in: *Proc. 3rd European Conference on Speech Communication and Technology (Eurospeech'97)*, Rhodes, Greece, 1997.
- [70] C. Wang and S. Seneff, Lexical stress modeling for improved speech recognition of spontaneous telephone speech in the jupiter domain, in: *Proc. 7th European Conference on Speech Communication and Technology (EuroSpeech'01)*, Aalborg, Denmark, 2001.
- [71] J. Williams, P. Poupart and S. Young, Partially observable markov decision processes with continuous observations for dialogue management, in: *Recent Trends in Discourse and Dialogue*, L. Dybkjaer and W. Minker, eds, Springer, 2006, pp. 191–217.
- [72] J. Williams and S. Young, Partially observable markov decision processes for spoken dialog systems, *Computer Speech and Language* **21**(2) (2007), 393–422, Elsevier.
- [73] J.D. Williams, The best of both worlds: Unifying conventional dialog systems and pomdps, in: *Proc. International Conference on Spoken Language Processing (InterSpeech'08)*, Brisbane, Australia, 2009, pp. 1173–1176.
- [74] S. Young, The statistical approach to the design of spoken dialogue systems, Technical report, CUED/F-INFENG/TR.433, Cambridge University Engineering Department, Cambridge, UK, 2002.
- [75] S. Young, J. Schatzmann, K. Weilhammer and H. Ye, The hidden information state approach to dialogue management, in: *Proc. 32nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'07)*, Honolulu, Hawaii, USA, 2007, pp. 149–152.
- [76] S. Young, J. Williams, J. Schatzmann, M. Stuttle and K. Weilhammer, The hidden information state approach to dialogue management, Technical report, Department of Engineering, University of Cambridge, Cambridge, UK, 2005.