INTERNATIONAL JOURNAL OF SPEECH TECHNOLOGY 8, 45–66, 2005

Knowledge-Combining Methodology for Dialogue Design in Spoken Language Systems

RUBÉN SAN-SEGUNDO, JUAN M. MONTERO, JAVIER MACÍAS-GUARASA,
JAVIER FERREIROS AND JOSÉ M. PARDO
Grupo de Tecnología del Habla, Dpto de Ingeniería Electrónica, UPM E.T.S.I. Telecomunicación,
Ciudad Universitaria s/n, 28040 Madrid, Spain
lapiz@die.upm.es
juancho@die.upm.es
macias@die.upm.es
jfl@die.upm.es
pardo@die.upm.es

Abstract. In this paper, we propose a strategy for designing dialogue managers in spoken dialogue systems for a restricted domain. This strategy combines several information sources intuition, observation and simulation, in order to maximize the adaptation within the system capability and the expectation of the user. These sources are combined by an iterative process consisting of five steps, where different dialogue alternatives are proposed and evaluated sequentially. The evaluation process includes different measures depending on the information required. Several measures are proposed and analyzed in each step. We also describe a user-modeling technique and an approach for designing the confirmation sub-dialogues based on recognition confidence measures. The knowledge-combining methodology is described and applied to a railway information system. In a subjective evaluation, users from the university gave the system a 3.9 score on a 5-point scale with an average call duration of 205 seconds. The employers of the railway company were more critical of the system. They gave it a score of 2.1 even though the system resolved more than half of the calls (57.8%) within an average call duration of three minutes (185 seconds).

Keywords: dialogue design, dialogue evaluation, confirmation subdialogues, user modeling, railway information system

1. Introduction

The dialogue manager is one of the most important components in a Spoken Language System. This module is responsible for managing the different resources provided by the rest of the modules: speech recognition, parsing, language generation, speech synthesis and database querying. Moreover, the dialogue manager is the module that defines the interaction with the user and it is the window through which the user perceives the system's capabilities.

The state of art in speech technology allows automatic systems to be developed that can work in real conditions. Telephone-based spoken dialogue systems have turned out to be an important field for applying these technologies. During the last few years, a significant number of spoken dialogue systems have been implemented for restricted domains. In Europe, we can highlight the Philips system (Aust, 1995), the ARISE project (Lamel, 2000; Baggia, 2000) and the BASURDE[LITE] system (Trías and Mariño, 2002) as examples of train travel information and ticket reservation services, and the Philips Directory-Assistance system (Schrâmm, 2000) as example of directory information service. In the USA, an important project has been the DARPA Communicator (http://fofoca.mitre.org), which has involved the most important research centers in this country: AT&T (Walker, 2001), MIT, BBN, Carnegie and Mellon University (Rudnicky, 2000; Carpenter, 2001), University of Colorado (Pellom, 2000; Zhang, 2001), Bell Labs, SRI and IBM (Gao, 2001). The main evaluation results have been presented recently (Walker, 2002a, 2002b). These services enable natural conversational interaction with telephone callers who access information on airline flights, hotels and car rental. Finally, it is necessary to refer to the AT&T call redirection system, "How may I help you?" (Riccardi, 2000), and the JUPITER system (Zue, 1997, 2000) (weather information) because theirs developers were pioneers in these systems. In Spanish, we can comment on the ACIMET system (Padrell, 2002) as an example of a weather information service.

In this paper, we present a methodology for designing the dialogue manager in a spoken dialogue system for restricted domains. We used this methodology to develop a train timetable information system for the main Spanish intercity connections. We also present an approach for designing the confirmation mechanisms and a proposal for incorporating user-modeling techniques in dialogue management. This paper provides a complete and detailed description of previous conference publications (San-Segundo, 2001a, 2001b, 2001c), including new experiments with updated results and new ideas obtained from the field evaluation carried out with final users from the railway company (RENFE). This paper also presents a review of the dialogue models considered in the international community, plus a short analysis of the evaluation problem in spoken language systems.

2. Background

Two of the most important challenges in dialogue design are dialogue modeling and dialogue evaluation. The next section discusses key issues in these areas.

2.1. Dialogue Modeling

In order to design a dialogue manager, it is necessary to define a model of the dialogue to be used as implementation architecture. The design process consists of obtaining the values for all the dialogue model parameters. From a high-level point of view, a dialogue model should include three basic elements/parameters:

- *Turn structure*: The data structure including all the information necessary for a dialogue interaction; e.g., language model and dictionary for recognition, prompts used by the system, and the item value to confirm (in a confirmation turn). The designer could define different types of turn. Some examples are prompting, confirmation, information providing and database querying. The information stored depends on the turn type. The dialogue turns can be prefixed at the design stage, or they can be generated depending on the dialogue flow.
- *Interaction point/dialogue state*: The data structure reporting on the current state of the dialogue at any point of the call. This structure should include information such as the turns executed up to this point, the obtained items, the confirmed items, and several measures of the interaction quality.
- *Executing turn algorithm*: The algorithm that defines the sequence of turns. If the turns are generated dynamically, this algorithm should define the rules for this generation.

Within the speech community, you can find several dialogue models. The following sections describes the most important ones.

2.1.1. Finite State Grammar Model. The dialogue model is based on a finite state grammar as shown in Fig. 1. We can identify three components of a dialogue model:

- *Turn structure*: Each dialogue state is associated with a different turn. In each state, we consider a data structure to store the information related to the turn. This information depends on the turn type associated with this state: prompting, confirmation, etc.

  It is necessary to clarify that, in this model, there is an information flow between different states (e.g., an item value). This information should be passed from the prompting turn to the confirmation turn. This flow must be made through auxiliary data structures, accessible from all turns (states).
- *Interaction point*: This point is defined by the last turn executed and the information stored in the previous turns. From these structures, we can determine the dialogue history and the interaction point.
- *Executing turn algorithm*: In this case, the algorithm is very simple. The dialogue flow is specified by the state order in the finite state grammar. When there are several output paths from the same state, the algorithm will check the corresponding question and make a decision.

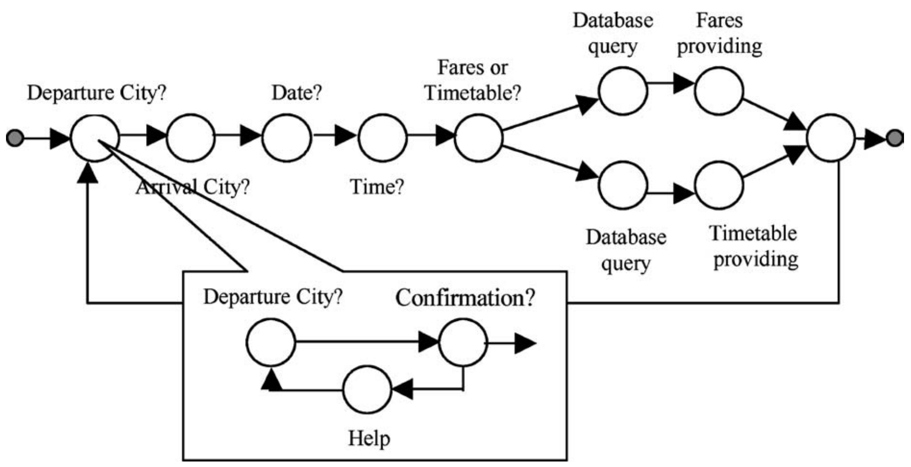


Figure 1. Finite state grammar for a train fares and timetable information system.

The main advantage of this model is its simplicity: when you define the grammar, the dialogue flow is fixed. On the other hand, this model presents a high degree of rigidity and does not allow mixed-initiative dialogue managers to be developed. Some examples of this kind of model are provided in Sutton et al. (1998), Baggia et al. (2000) and Córdoba et al. (2001).

2.1.2. Frame-Based Model. In this case, the dialogue model is based on frames. For every goal in the dialogue, we define a frame associated to it. A dialogue goal is part of the functionality provided by the service: fares or timetable information, ticket reservation, etc. Every frame contains the items needed to satisfy the goal (data necessary to query the database and to obtain the information requested). In Fig. 2, we represent the two goals (FARES and TIMETABLE) considered in the example of Fig. 1.

In Fig. 2, we also show the example fields contained in the "Departure City" structure (Timetable Frame): its name, the value recognized, the system prompt, the confirmation prompt and a boolean parameter reporting the item confirmation. The dialogue model components are as follows:

- *Turn structure*: In this model, the turn structure is dynamic and it is generated depending on the dialogue flow. At every turn, the system creates a turn structure containing the information needed for this interaction. This information comes from the item structures in the goal frame.

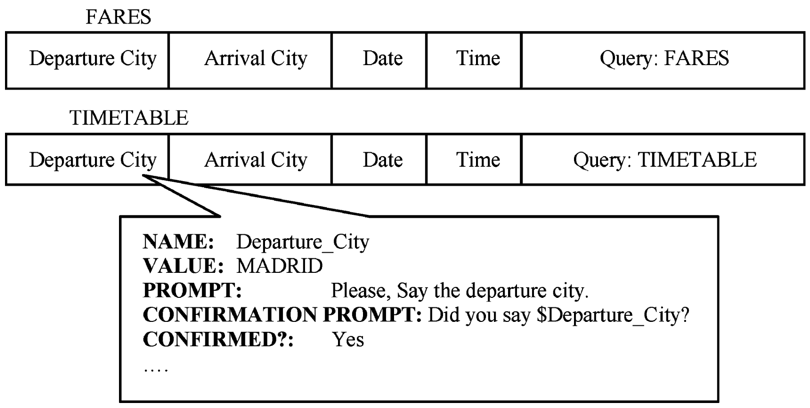


Figure 2. Frames representing the goals shown in Fig. 1: FARES and TIMETABLE.

- *Interaction point*: We can define the dialogue state by checking the information contained in the frame structures. In these structures, we have the item values which were confirmed as well as the items that are empty. When the system obtains and confirms all the items, the goal can be satisfied (the system accesses to the database and provides the required service). We can also store all the turn structures generated during the interaction. We use this set of structures for debugging or for aiding/supporting the decisions of the executing turn algorithm.
- *Executing turn algorithm*: In this case, the algorithm is more complex because it creates turn structures dynamically. At the beginning, the system must detect the goal required by the user (and the associated frame). From this point, the algorithm generates the turn structures, attempting the following actions in the priority listed (Ward and Pellom, 1999):

  o Clarify if necessary. If the interpretation is uncertain or ambiguous, interact with the user to get a unique and high-confidence interpretation.
  o Retrieve data and present them to the user. If there is enough information to build a data query, build it and get the results.
  o Prompt user for required information. If there is not enough information to take the desired action, prompt the user for the required information. The rules for deciding what to prompt are very simple because we define an item priority. We always select the highest priority unfilled slot. Another alternative is to develop rule-based systems where several rules define the next item to ask, considering the interaction point (Litman, 2002).

This model is more complex, but it is more flexible and permits mixed initiative (user and system initiative) dialogues. The user can change the order of goal or item specification. Although this model is more powerful than the previous one, they are not incompatible models. In a dialogue module both strategies can coexist. Sometimes, there are complex sub-dialogues that require a full system initiative strategy. For these cases, the best way of modeling the dialogue is by using a finite state grammar. For the rest of the dialogue, we can use the frame-based model. Some examples of this kind of model are given in Aust et al. (1995), Goddeau et al. (1996), Bennacef et al. (1996), and Constantinides et al. (1998).

2.1.3. Hierarchical Model. The hierarchical model is similar to the frame model. The main differences correspond to the structures used for dialogue representation. In this case, we use a hierarchical structure, where we show the main goals of the services, their sub-goals and the items associated with each sub-goal. In Fig. 3, we show the hierarchical structure for the example used during the exposition.

The highest level includes the goal structures while the lowest level contains the item structures. These data structures are similar to those described in the previous section (frame and item structures). In this case, a new intermediate level is incorporated for increasing flexibility: the sub-goal level. The main sub-goal characteristics are as follows:

- The sub-goals permit several items to be grouped, defining dialogue steps (sub-dialogues). Between these steps, we can define rest zones, where the

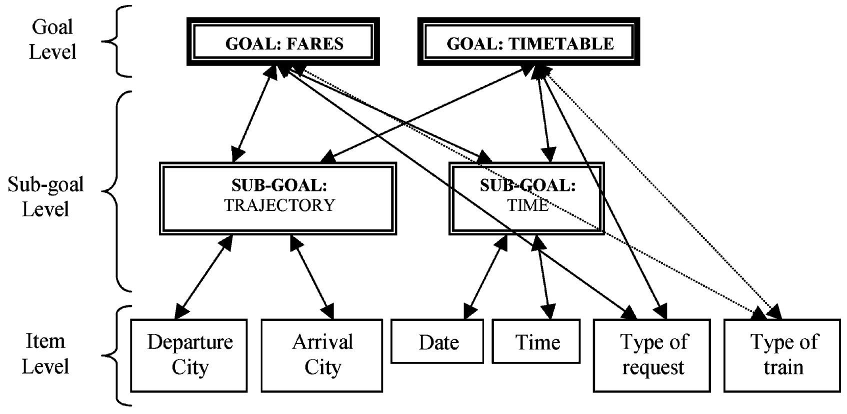


Figure 3. Hierarchical structure representing the goals shown in Fig. 1: FARES and TIMETABLE.

system can offer help to the user or make an interaction summary. The system can control the dialogue rhythm.

- The sub-dialogues permit the complexity of prompting a set of items to be encapsulated. For example, for the trajectory sub-goal (including departure and arrival cities), we could consider both items as independent: each item is prompted and confirmed in different turns. On the other hand, it is possible to prompt or confirm both items in the same turn, saving dialogue turns and time. Independently of the item prompting, the goal and sub-goal structures would be invariant.
- From the previous point, we can deduce that sub-goal nodes have structures similar to those contained in item nodes. These structures contain fields like the sentence to prompt all items at the same time, the sentence to confirm all items simultaneously and the confirmation flag (this flag will be true when all the sub-goal items have been confirmed).

The structures at the goal level are composed mainly of flag fields: a flag to indicate that the goal has been selected or a flag to inform that all the items have been obtained from the user and it can query the database.

In the hierarchical model, it is necessary to define relationships between fields contained in goal, sub-goal and item structures. For example, a sub-goal should be considered confirmed when all the items associated with it have been confirmed. And vice versa, when one sub-goal has been confirmed, all the items should be tagged as confirmed. These rules define the integrity of the dialogue state. When the state of the dialogue is modified, all the relationships should be checked, and all the field-related structures should be updated.

This model permits optional items to be defined (an example was shown in Fig. 3 with a dash line: the type of train). An optional item is not necessary to access the database, so it is not prompted. If the user wants to specify it, the system will collect it, constraining the database query. Considering the integrity rules previously discussed, we can extend the optional item concept. Through these rules, we can define logical functions among several items, leaving the choice to the user. An example is presented in Fig. 4.

In this example, the sub-goal trajectory will be completed when the user specifies the departure and arrival places. A place is specified by the city name[1] or by the station name. In this case, the item type (optional/mandatory) depends on the user response. The

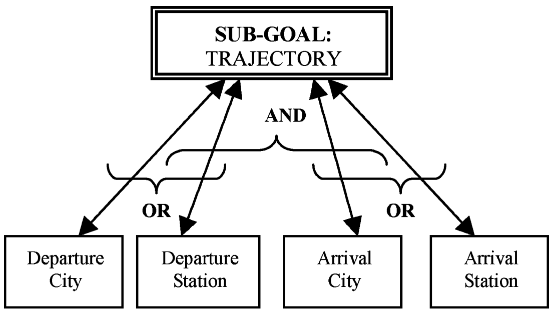


Figure 4. Example of an extended concept of optional items.

user decides what to say the city name or the station name. This technique offers higher flexibility.

On the three components of the dialogue model, the comments are similar to those made in the previous section. The turn structures are generated dynamically and the dialogue state is defined by the content of the hierarchical structure. The dialogue sequence is defined based on the priority of the items in each sub-goal, and the sub-goal priority in each goal. As we commented before, rule-based systems are an alternative to define the dialogue flow when the user does not take the initiative. The user can change the prompting item sequence or the focused goal (mixed-initiative). As the dialogue advances, the hierarchical model can change its structure, adding new sub-goals, deleting old sub-goals or modifying the item characteristics their priority, optional/obligatory aspect, etc.

Examples of this dialogue model include some of the dialogue managers generated in the Darpa Communicator project (Ward and Pellom, 1999), Pellom et al. (2000), Rudnicky et al. (2000), Zhang et al. (2001), Carpenter et al. (2001), Walker et al. (2001), and Gao et al. (2001).

As we can conclude from the dialogue model descriptions, the principal elements of a dialogue model are goals and items. Other important aspect is the item asking sequence (dialogue flow) for obtaining the information necessary to satisfy a goal. In the methodology description, we will focus the design effort over these aspects.

2.2. *Dialogue Evaluation*

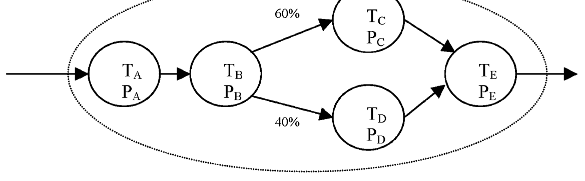
Any human-computer interaction can be represented by a sequence of states (or dialogue steps). A dialogue step includes one or several turns related to it, prompt and confirmation turns, for example. These steps can be prefixed or can be generated dynamically. For evaluating each dialogue step, it is necessary to consider these two parameters:

- Average time needed to complete the step T_{AVG}.
- Error probability P_{ERROR}. This is the probability that the user will not complete the step and will finish the interaction with the system (hang up the telephone).

Depending on the step complexity, we can obtain an estimate of these two evaluation parameters. This estimate can be made by considering measures such as the time needed to prompt an item, the average response length, the recognition error rate, etc. When the step contains many turns, the parameter estimation is more complex, and it is necessary to consider real user interactions in order to measure these parameters (Suhm and Peterson, 2002).

When defining the dialogue steps, we can consider different granularity levels. We can define dialogue steps that group several smaller steps. In this case, we calculate T_{AVG} and P_{ERROR} knowing the parameters of the smaller grouped steps and the combining structure. See the example in Fig. 5. We show a big step that includes five small steps (A, B, C, D and E), and its parameter calculation.

Extending this idea, we can establish a relationship between the parameters of the small steps, the sub-dialogues (bigger dialogue steps) and the whole dialogue. These parameters permit the dialogue quality at several levels to be measured. In dialogue design, several dialogue strategies should be tested and evaluated. The best option should consume the lowest average time with the lowest error probability. This idea has been considered as an important criteria during the designing process to select the best dialogue alternative. One problem appears when the option with the lowest average time has a higher error probability,



$$T_{AVG\,TOTAL} = T_A + T_B + 0.6T_C + 0.4T_D + T_E$$
$$P_{ERROR\,TOTAL} = 1 - [(1-P_A)(1-P_B)\{0.6(1-P_C) + 0.4(1-P_D)\}(1-P_E)]$$

Figure 5. A complex step example.

or vice versa. In this case, it is necessary to compute a unique measurement obtained from these two parameters. Levin and Pieraccini (2000) propose using a linear combination of both parameters. For this formulation, the problem is how to calculate the weight for each parameter. These weights are not unique because they depend on the type of service offered. Let us consider two different situations: the first one with $T_{\mathrm{AVG}} = 4.2$ min and $P_{\mathrm{ERROR}} = 40\%$ and the second one with $T_{\mathrm{AVG}} = 8.1$ min and $P_{\mathrm{ERROR}} = 20\%$. Considering a service where there is human operator support for difficult interactions, we would choose the alternative with lower T_{AVG} even though the P_{ERROR} is higher. In this case, the incorrect interactions will be managed by a human operator, so we do not pay attention to the P_{ERROR} except if it is excessive high. On the other hand, if there is no human operator support, the best alternative is the one that offers a lower P_{ERROR} even though the T_{AVG} is higher. Because of this, the dialogue evaluation (during design) and dialogue testing for a specific service must consider the service characteristics (relevance of the information provided by the service), its competency (other similar services provided by a human operator or automatic systems) and user expectations.

Hacioglu and Ward (2002) propose a strategy for calculating the combination weights by analyzing previous user tests. Their strategy is based on the correlation between user satisfaction and the measures obtained during the testing. This proposal is a modified version of the PARADISE strategy Walker et al. (1997, 1998). Minker (2002) presents an overview of recent activities in spoken dialogue system evaluation.

3. Methodology for Dialogue Design

The methodology proposed in this paper consists of five steps. The first step is the database analysis in which the information contained in the domain database is described by an Entity-Relationship diagram (E-R). In design by intuition step, a "brain-storming" on the E-R is carried out for proposing different dialogue alternatives. The third step is design by observation, where we evaluate each proposal using user-operator dialogue transcriptions. In the next step, we simulate the system with a Wizard of Oz design in order to learn the specific characteristics of human-system interactions. The fifth step is the design by iterative improvement, in which the confirmation strategies are designed in order to implement a fully automatic system. In this step, we also propose a user-modeling technique. Although the methodology is described sequentially, new dialogue characteristics can appear at any step, making it necessary to re-define and re-evaluate some aspects of previous steps. Because of this, the methodology should be interpreted as an iterative process consisting of five design steps.

The railway information system developed in this work uses isolated speech recognition, but the methodology is general and can be used to develop systems with continuous speech recognition and understanding.

This methodology is similar to the Life-Cycle Model presented in Bernsen (1998) and (www.disc2.dk), but we incorporate the step "design by observation" in which human-human interactions are analyzed, and we present measures to evaluate the different design alternatives at every step of the methodology.

4. Steps 1 and 2: Database Analysis and Design by Intuition

The database, as the initial point in our methodology, contains the information or knowledge we want to provide in our service (the information that the system will manage). For example, in a railway information system, this database contains the timetable information for all the trains, their prices, their services, etc. The aim of the database analysis is to describe this information. This description consists of an Entity-Relationship Diagram (E-R) that shows a semantic representation of the data (Fig. 6). In this diagram, the main entity sets, their attributes and keys (attributes that uniquely define an element in an entity set), and entity set relationships must be defined. The E-R diagram is the domain representation (data representation) that end users will perceive through the service. Because of this, the E-R diagram must match user expectations perfectly. In

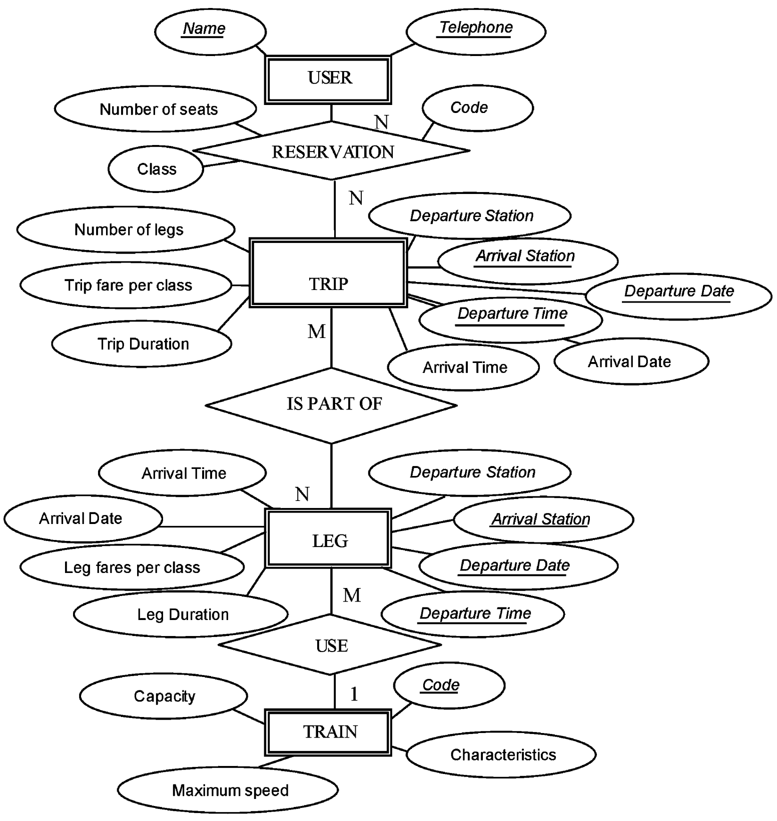


Figure 6. Entity-Relationship diagram defined for our railway information service.

order to guarantee this, end users should collaborate with the designers in the E-R diagram definition. The E-R diagram is not unique and depends strongly on the system designers, the end users and the service provided. Sometimes, the E-R diagram can be very different from the database structure. In these cases, it is necessary to develop a new module that translates the original database structure to the E-R diagram view.

The entity sets are critical because they will be the possible goals that the system can provide (parts of service such as timetable information, reservations, fares, etc.). The entity sets that contain a great amount of data will be especially interesting. They will contain a high percentage of data that the system will manage, as such they will probably be requested frequently by the users. On the other hand, the keys are the main mandatory items for accessing entity elements. Because of this, the user must be asked about them to define the dialogue interactions. A provisional order of prompting can be drawn by considering entropy criteria. The items that produce a higher information increment should be queried first. This way, the user perceives the system as a rapid way to glean the desired information.

After the database analysis, we propose "brain-storming" (design by intuition) on the E-R to propose dialogue *alternatives*. These proposals are concerned with the goals to be provided, the sequence of offering them, the items needed to satisfy each goal (pieces of information that must be obtained from the user, such as departure and arrival cities, departure date, etc.), and the sequence of asking them. As we commented in Section 2.1, these are the most relevance aspects in dialogue design. Other aspects like the ways the user can specify each item can be considered. For example, for the place of departure, the user can say the station name or the city name. In the second case, the system should provide the information for all the train stations in that city. Expert designers and final users should be involved both in the "brain-storming" and the E-R diagram definition. The result of this analysis is a table or worksheet that includes all the alternatives. Although, these two steps do not require significant design effort, they have the disadvantage that the experience of a small group of people is not always enough to capture all variability in user behavior. Because of this, the alternatives presented in the worksheet should be validated and completed in the next steps. This table will be used in the next methodology step to compute the frequency of each alternative concerning the goals, the information that the system must provide, the ways the user specifies an item, etc.

5. Step 3: Design by Observation

Design by observation is intended to analyze user-operator dialogues in a similar service and track off the observed events. This design phase evaluates and measures the impact of the alternatives proposed at the previous step. The relevance of the alternatives is calculated by counting how many times they appear in the user-operator dialogues. New alternatives can be found during this analysis and must be included. These new alternatives can redefine the E-R diagram, making it necessary to repeat partly the design by Intuition process. This example illustrates that this methodology should not be interpreted as a rigid sequential process but an iterative one. The analysis carried out in the design by observation step is focused on three aspects: goals, items and negotiation.

5.1. Goals Analysis

In the goal analysis, we focus the study on two aspects:

(a) *Goals that are most frequently required by the user, and the sequence of querying about them.* The result of this analysis is the number of times that a specific goal is required and its position in the goal sequence of the call. In Table 1, the goal analysis for 100 call transcriptions is presented. Several goals can appear in the same call. In our final system, we offer the goals that appear in more than 10% of

Table 1. Goal analysis.

		Position				
	%	1st	2nd	3rd	4th	5th
Timetable	64	57	6	1	–	–
Round trip timetable	20	–	14	5	1	–
Fares	46	6	30	10	–	–
Make a reservation	26	14	4	2	3	3
Train frequency	2	2	–	–	–	–
Itinerary	14	8	4	1	1	–
Offers	5	1	2	1	1	–
Others	12	8	2	–	2	–

calls, except *Itinerary* (there is no information in the database) and *Others* (it contains very different goals but none reaches 10%).

From this table, we conclude that the timetable information is a very important goal, and it is usually asked for at the beginning of the interaction. The "fares" goal is also an important one that appears in second position, similar to the "round trip timetable" goal. Generally, we have observed two kinds of interaction. In the first one the user asks for timetable information and then for fares or round trip information. The second kind shows a well-informed user who calls the service to make a reservation directly ("to make a reservation" appears as the first goal in a significant number of calls). This analysis shows a pattern in the behavior pf Spanish users. They call the service mainly to get information (timetable and fares). After the first call, some of them phone again to make the reservation.

(b) *The information given by the operator to satisfy each goal.* We make a note of these data (e.g., departure and arrival times, trip duration, and train type) and their importance (number of times that the operator provides them). We also make a note of the order for providing information (if it is fixed for all cases or it depends on the data provided). These aspects are related to the way the human operator summarizes the information, selecting the most relevant data and the exposition order.

5.2. *Item Analysis*

To satisfy a specific goal (part of a service), the system must request some items from the user in order to get the information necessary for the required action. These questions also must be studied in step five. The aspects to be analyzed are as follows:

(a) *To determine which items are needed to satisfy each goal and the sequence for quering them.* For studying this aspect, it is necessary to make note of how many times the user provides each item (or the human operator asks for it) and its order: which items are specified/required during the first turns and which ones in the last turns.

(b) *To classify items as mandatory or optional.*

- *Mandatory.* these are items needed to satisfy the goal and the user must be queried about them (e.g., "What date are you departing?"). If the user does not provide this information, it must be assigned a default value (e.g., "today") or the goal have to be satisfied depending on the item value (e.g., for all days in current week).
- *Optional.* the system does not ask the user for it. When the user specifies it, the provided information should match the item value.

As previously mentioned on, a first item classification was obtained from the database analysis. In this case, we considered the key attributes as the mandatory items, and the rest as optional ones. The user can provide optional items to constrain the service required. During the observation analysis, this classification can change slightly because some mandatory items are re-classified as optional ones. This phenomenon occurs when the human operator wants to offer several options (instead of just one) to help the user decide (e.g., in our case, the human operator asks for the departure date but not for the departure time. The operator tries to provide several options to the user who will decide the best time depending on schedule preferences).

(c) *To classify each item as simple or complex.* We consider an item as complex when it can be divided into several simple ones (e.g., "departure date" can be divided into DAY, MONTH and YEAR). This classification depends on the available speech technology. With isolated speech recognition, each simple item requires one interaction; a continuous speech recognizer could solicit several simple items simultaneously.

(d) *To analyze the different ways a user can specify an item value and its importance.* At this point, it is necessary to make note of the different ways of item specification and the percentage of times each way is considered.

Table 2 shows the results for the departure date. A high percentage of people travel during the same week that they are inquiring and a relevant percentage do not specify any date, because they just want general travel information between two cities. 9.4% of people traveled during the same month and

Table 2. Analysis for the departure date.

Current week		This month	Other month	Any
Today	25.0%	9.4%	4.7%	28.1%
Tomorrow	15.6%			
Weekday	17.2%			

specified the day by stating just the date of the day (e.g., the 20th). 4.7% of people traveled on another month and specified the date by the name of the month and the day.

One noteworthy aspect is the high percentage of users who do not specify the departure date (28.1%). This effect shows a Spanish habit in train information requests. Frequently, when they phone the operator, they have not yet decided the departure date. They want general information to plan a trip. In these cases, a mandatory item is not specified, so the human operator must propose an item value as default (e.g., today) or provide the information for several item values (e.g., for all the days in the current week, or for two days: week and weekend day).

Another required query is the place of departure, approximately 90% of the users specify the departure place with the name of city, and only 10% use the station name. This analysis is very important to properly define the recognition vocabularies and language models.

(e) *Item ordering and grouping*. One step in the dialogue is a set of user-system interactions in which the user could be asked several items without confirmation. Our recommendations are:

- Grouping items that are contiguous in the item sequence and have a semantic relationship between them (e.g., departure date and time).
- Not to group more than four consecutive items.

Just one confirmation can validate a group of items. We can also incorporate rest zones between consecutive groups to provide help or an interaction summary. The dialogue steps impose a certain rhythm to the user-system interaction.

5.3. Negotiation Analysis

The "negotiation" is the process through which the user must accept or reject the system proposals, by being able to change the constraints and modify them. The target of the negotiation sub-dialogue is to match the system alternatives to the user's needs. In order to study this type of dialogue it is necessary to evaluate the following aspects:

(a) *The information that helps the user to make a decision, and its importance.* We have to annotate each different criterion used during the negotiation (e.g., departure date, departure time, fares, travel duration, number of connections) and the number of times it is considered (criterion relevance).

Table 3. Criteria analysis for negotiation.

Criteria	%	Criteria	%
Departure time	41.0	# Connections	2.6
Arrival time	15.4	Connection place	2.6
Departure station	2.6	Class	10.3
Fares	7.7	Duration	5.0
Train type	12.8		

Table 3 shows the analysis for negotiation criteria. As shown the most used criteria are the departure and arrival times. The number of connections is an important factor, but it does not appear frequently in user-operator dialogues because we have observed that the operator only offers direct trips without connections. The operator only offers trips with connections when there is no direct trip, in these cases, it is an important factor.

Another interesting result is that Train Type is more frequently used for negotiation than Fares. This effect occurs because there is a relationship between these two aspects, known by the users. And the Train Type offers more information, it provides a hint not only about fares but also about the services offered on the train.

(b) *To choose the negotiation strategy.* There are two negotiation strategies: the system can present the best option and let the user ask for the previous/next one (navigation), or the system can present several alternatives at the same time and ask the user to choose one of them (or select another set of alternatives if none matches the user's needs). In this case, it is important to analyze the *number of alternatives* the operator gives simultaneously (and that the user can manage), during the conversation.

In the observation analysis, only human-human interactions are considered, and they can be extremely different from human-system ones (Doran et al., 2001). Because of this, high level dialogue characteristics can be learned, but specific behaviors when interacting with an automatic system are not detected (e.g., changes in the user speaking rate, and usage of formal vs. colloquial phrases). On the other hand, the observation analysis permits several alternatives to be evaluated without having to actually implement any system. The observation step is not always possible because there is not always a user-operator system with the same characteristics as the automated system we want to design. In these cases, we have to choose between two options. The first one consists of defining a user-operator service and evaluating the dialogue alternative with user tests. The second option is to skip this step and proceed to the design by simulation. In this case, the simulation complexity (Step 4) increases because a larger set of alternatives must be evaluated.

6. Step 4: Design by Simulation

In step four, the specific characteristics of human-system interactions are analyzed by simulating the system with a Wizard of Oz approach. The dialogue alternatives proposed in the previous steps of the methodology are now implemented and evaluated through a simulated system. We must focus on the dialogue flow design, the prompts for obtaining the necessary items, and the information provided by the system to satisfy each goal. We have to pay attention to those aspects where the conclusions derived through intuition mismatch the observation analysis. The Wizard of Oz strategy bases its efficacy on the users' belief that they are interacting with a fully automatic system. In other case, this tool has not use.

In our evaluation, 15 users called a simulated system, completed six scenarios and filled in a questionnaire. The evaluation measurements came from system annotations (referred to as the *System*) and from user answers to the questionnaire (referred to as the *Questionnaire*), where the user provided answers to subjective aspects not easily measured by the system. The user speech was recorded in audio files. The aspects to be evaluated and the evaluating measures are described in the next sections.

6.1. Goal Evaluation

The main aspect in the goal evaluation is *the goal sequence and the goal coverage.* The evaluating measures are:

- *System*: The number of times a goal is required by the user, time and number of questions (interactions) needed to satisfy a goal.
- *Questionnaire*: New goals suggested by the user.

6.2. Item Evaluation

In the item evaluation, we should evaluate four aspects:

(a) *Design of the questions and recognition vocabulary.* The measures are:

- *System*: The number of times the user remains silence (the user does not answer the system question) and the recognition rate for each question (if a first version of the recognizer is already available). These measures provide information on the question/prompt intelligibility and its suitability within the dialogue flow.

  All the user speech is recorded in audio files. The analysis of these files provides information on how the user interprets the system prompts and helps us to define the first version of the recognition vocabularies.
- *Questionnaire*: How easy is it to specify an item value and how comprehensible are the system questions.

(b) *Item sequence evaluation.* The method to evaluate different dialogue flows (item question sequences) consists of proposing several sequences of questions and randomly selecting one in each call. In order to compare the different alternatives, we consider the following measures:

- *System*: When possible, the Sequence Recognition Rate (SRR) computes the product of all independent item recognition rates (if there is a first version of the recognizer available).

  Table 4 shows the sequence analysis for the departure and arrival cities. The SRR difference is not significant, but we observed the following effect: when the system asks the Arrival City first, the user assumes that the system knows the Departure City (city where the user is calling from). When the system finally asks for it, the user gets confused. This explains the great

Table 4. Sequence evaluation for departure and arrival cities (item recognition rate and sequence recognition rate).

	Recognition rates (%)		
Sequence	1st item	2nd item	SRR
Departure-arrival city	75.6	80.0	60.5
Arrival-departure city	94.3	60.0	56.6

difference between the item recognition rates for the Arrival-Departure City sequence.

As in the goal analysis, the system also must make a note of the time and number of questions (interactions) needed to get all the item values from the user.

- *Questionnaire*: Asking the user for his/her preference.

(c) *Analysis of different ways to split complex items into simple ones.* This aspect is very similar to that previously described. In this case, the target is to design the best simple item sequence. This process must balance two aspects: the number of simple items (sequence length) and the ease of specifying each simple item (keeping in mind the speech technology available and the complexity of the vocabulary). The measures are similar to those described previously:

- *System*: Number of interactions and time to obtain a complex item, and the Sequence Recognition Rate for the simple items.
- *Questionnaire*: How easy is it to specify a complex item with the proposed sequence and how familiar the simple items are to the user.

(d) *Managing the mandatory items when the user does not specify any value for them.* We have two alternatives: to parameterize the information by this item (satisfy the goal for all possible item values) or fix a default value and constrain the information provided. The response for this disjunctive depends on the user's habits, so the solution must come from the human operator experience or from the user's comments in the questionnaire.

- *Questionnaire*: We ask the user in the questionnaire whether the information given by the system exceeds his/her expectations (and it is necessary to fix a default value) or whether it is adequate.

6.3. Negotiation

In our case, we have decided to present several travel options at the same time and ask the user to choose one of them. In this solution, we need to evaluate two aspects: the number of options presented at the same time and the information provided for each travel option. Randomly, the system presents the options one by one, two by two, or three by three. For the information provided per option, several patterns are designed and randomly selected for each group of options. The measurements considered for evaluating the different alternatives are:

Table 5. Negotiation analysis in the simulation step.

Negotiation analysis (User questionnaire)		
1 by 1	2 by 2	3 by 3
21.4%	21.4%	57.2%

Negotiation Criteria				
Train type	Depart. time	Arrival time	Fares	Trip duration
15.6%	37.5%	25.0%	15.6%	6.2%

- System: Number of questions and negotiation time.
- Questionnaire: What information helps the user to choose, and what number of options can he/she manage at the same time?

In Table 5, we show the results. Users preferred to manage three options at the same time; there are fewer interactions but the negotiation takes more time. Because of this, in our implementation, we decided to limit the negotiation in a three by three basis but reduce the information provided per option to Train Type, Departure Time and Arrival Time.

For overall dialogue evaluation, we propose the following measures:

- *System*: Average number of interactions, time per call, and recognition rate for all the vocabularies.
- *Questionnaire*: Users must evaluate on a scale how fast they obtained the information, how easy it was to learn the service, which part they would change, and how they compare the system to other ways of obtaining this service: web, going to the station, etc.

Whenever possible, it is better to use only measures computed by the system because they are objective. The questionnaire cannot be longer than one sheet and should only address issues that cannot be resolved by automatic system annotations, such as subjective evaluations or default values for mandatory items. If we want to include many questions, it is better to design different questionnaires evaluating different aspects in each. Every user should complete only one of the questionnaires proposed. Regarding the scenarios to complete, it is important that the users define their own scenarios for testing a greater diversity of situations. Some scenarios should be imposed when sufficient data are required to evaluate specific situations.

The Wizard of Oz method permits us to analyze different dialogue designs with no fully automatic system implemented. Moreover, the system does not need to confirm the item values, so we can make independent designs for the dialogue flow and for the confirmation mechanisms. On the other hand, the main problem in this step is the difficulty simulating an automated system, especially the reply time. For example, a person (wizard) spends more time in selecting an item value from a list than an automatic speech recognizer (although the recognizer does not always pick the right value). It is necessary to develop tools that help the wizard to simulate the system in a realistic way.

7. Step 5: Design by Iterative Improvement

In step five, we implement the first version of a fully automated system in an iterative modify-and-test process. With design by simulation (Section 6), a first version of the dialogue flow was defined, and now it is necessary to design the confirmation mechanisms. These mechanisms have a relevant influence on the general performance of the system. Good confirmation mechanisms can avoid asking the user questions to validate the recognized words, thus making the dialogue faster. These mechanisms take into account the recognition confidence: an analysis of confidence measures is required. On the other hand, when the system does not confirm some words because they have high recognition confidence, this assumption can fail. Because of this, fast error recovery techniques are required. In this step, we propose different error recovery techniques and describe a user-modeling approach for our railway information system.

7.1. Confirmation Mechanisms

To design the confirmation mechanisms in spoken dialogue systems, it is necessary to describe the confirmation strategies available, obtain confidence measures from the recognition module, and define a relationship between confidence values and confirmation strategies (Sturm, 1999).

7.1.1. Confirmation Strategies. Confirmations strategies vary in terms of:

1. the number of items to confirm:
   - One item . ("Did you say Madrid?")
   - Several items. ("Do you want to go from Madrid to Sevilla?")
2. the possibility to correct (Lavelle, 1999):
   - *Explicit confirmation*: The system confirms one or several item values through a direct question. ("I understood you wanted to depart from Madrid. Is that correct?")
   - *Implicit confirmation*: The system does not explicitly permit the user a correction; it only reports the recognition result. ("You want to leave from Madrid. Where are you going to?")
   - *Semi-implicit confirmation*: This is similar to implicit confirmation, but the user can correct ("You want to leave from Madrid. In case of error, say correct, otherwise, indicate your destination city"). This confirmation allows error recovery (Lavelle, 1999), but it is not very friendly for the user. In Section 7.1.6, we describe the CORRECT command that permits the same functionality without increasing the prompt.
   - *No confirmation*: The system does not provide feedback on the recognized value (e.g., in yes/no questions).
   - *Item value rejection* and repeat question: When the confidence is low, the system does not present the value to the user and repeats the question ("Sorry, I could not understand. Where are you departing from?").

7.1.2. Confidence Measures in Speech Recognition. Confidence measures in speech recognition are very useful for designing the confirmations (Sturm, 1999). The recognition module used in our system is a large vocabulary telephone speech recognizer that can recognize isolated words and simple expressions such as "On Monday", "Next week" or "In the morning". The recognizer (Macías, 2000a) is based on a hypothesis-verification approach. The best features for confidence annotation are concerned with the verification step and are based on Macías et al. (2000b):

- *First candidate score*: Acoustic score of the best verification candidate.
- *Candidate score difference*: Difference in acoustic score between the 1st and 2nd verification candidates.

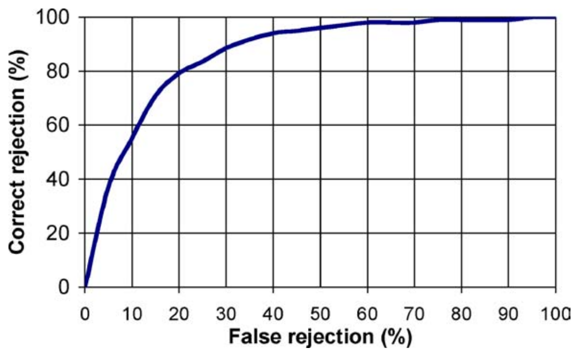


Figure 7. Correct rejection vs. false rejection.

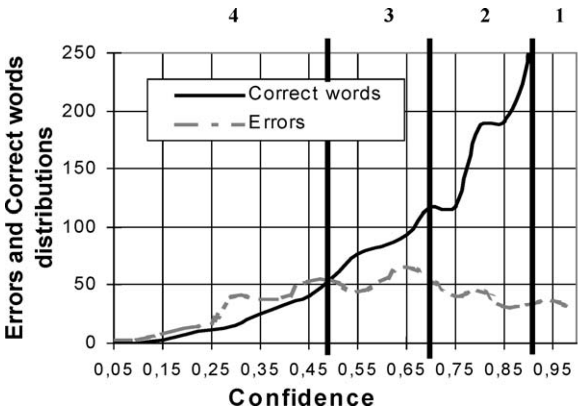


Figure 8. Error and correct word distributions vs. confidence.

- *Candidate score mean and variance*: Average score and variance on the ten best candidate names.
- *Score ratio*: Difference between the score of the phone sequence (hypothesis stage) and the score of the best candidate word (verification stage).

All the features are divided by the number of speech frames throughout the utterance. We have considered a Multi-Layer Perceptron (MLP) to combine the features in order to obtain a single confidence value. In this case, we use the features directly as inputs to the MLP. With this solution, preprocessing is required to limit the dynamic range of each measure to the (0,1) interval. Here, the normalization scales the features using the minimum and maximum values obtained for each measurement in the training set. The hidden layer contains ten units, and one output node was used to model the word confidence. During weight estimation, a target value of 1 is assigned when the decoder correctly recognizes the name, and a value of 0 signifies incorrect recognition. The database used in the confidence experiments is built with the results obtained in the recognizer evaluation for a 1,100 name dictionary. In this case, we have 2,204 examples, considering 1,450 for training the MLP, 370 as the evaluation set and 370 for testing. We have repeated the experiments six times providing a 6-Round Robin training to verify the results. In this paper we present the average results of these experiments. A 39.1% of wrong words are detected at a 5% false rejection rate (Fig. 7), thus reducing the minimum classification error from 15.8% (recognition error rate) to 14.0%.

7.1.3. Confirmation Mechanism Design. To design the confirmation mechanism, it is necessary to plot the correct word and error distributions as a function of the confidence value. For several years, there has been interest in defining a mathematical model for confirmation mechanisms (Niimi and Kobayashi, 1995, 1997). These models become more complex as the confirmation sub-dialogue complexity increases, making them computationally expensive and useless in design.

In this section, we propose an approximation that makes the design process easier. As shown in Fig. 8, we plot the correct word and error distributions, and we define different thresholds for splitting the plot into several confidence areas.

We define 4 levels (3 thresholds) of confidence (Fig. 8):

1. *Very high confidence*: The number of correctly recognized words is much higher than the number of errors. For example, the percentage of correctly recognized words is more than 95% of the recognitions (the percentage of errors is less than 5%).
2. *High confidence*: The number of correctly recognized words is higher than the number of errors: the percentage of correctly recognized words varies between 70 and 95% (the percentage of errors varies between 30 and 5%).
3. *Low confidence*: Both distributions are similar. The system is not sure about the correctness of the recognized word. In this case, the percentage of correctly recognized words varies between 50 and 70% (the percentage of errors varies between 50 and 30%).
4. *Very low confidence*: In this case, there are more errors than correctly recognized words, so we reject the recognized word and the system must ask again. The percentage of correctly recognized words is lower than the percentage of errors.

For the departure and arrival city steps, we define CL(D) and CL(A) as departure/arrival city Confidence Levels (CL). Depending on the CL, we implement the following confirmation strategies:

- CL(D) = 1 and CL(A) = 1: *Implicit confirmation of both items*. "You want to travel from Madrid to Sevilla, when do you want to leave?"
- CL(D) = 2 or CL(A) = 2: *Explicit confirmation of both items*. "Do you want to travel from Madrid to Sevilla?"
- CL(D) = 3 or CL(A) = 3: *Explicit confirmation of each item*. "Do you want to depart from Madrid?"
- CL(D) = 4 or CL(A) = 4: *Rejection of each item*. "Sorry, I do not understand. Where are you departing from?"

Under difficult conditions (due to noise, the use of out-of-vocabulary words, a novice user, etc.), the system asks the user to spell the city name (San-Segundo, 2002). In this design, we have not considered the semi-implicit confirmation because it is not very user friendly. In Section 7.2.1, we describe the correction command that permits the same functionality without increasing the length of the prompt.

7.1.4. Confirmation Sentences Design. Another important aspect in confirmation design is the set of sentences used for prompting. With careful sentence design, it is possible to implement different confirmation strategies by reusing a high percentage of sentences. Below are recommendations on sentence design.

(a) *Implicit confirmation*: In this kind of confirmation, the sentences typically consist of two parts: the system presents the value of the item and then, queries the new item (e.g., "You want to travel from Madrid to Sevilla, When do you want to leave?"). This structure permits an independent design to be developed for confirmation sentences as well as item questions. When there is a semantic relationship between the item to be confirmed and next item, we can join both sentences together, thus creating just one. For example:
S: "Which month do you want to leave?"
U: "July"
S: "Which *day of July* do you want to leave?"
Joining sentences does not permit independent design, but it produces a shorter dialogue.

(b) *Semi-implicit confirmation*: The structure is made up of three parts: the item presentation, the command for correction and the prompt for next item. The sentence is usually very long and unfriendly.

(c) *Explicit confirmation*: The structure consists of one sentence in which the system asks explicitly whether the recognized item value is correct or not ("Do you want to leave on July the 18th?"). This sentence can be divided in two; the first one presents the item value, and the second one asks about the correctness of the value ("You want to leave on July 18th. Is that correct?"). In this case, as for implicit confirmations, independent analysis is possible.

In order to make the system user friendly, the confirmation sentence must be designed based on the user answer. If he/she says "*this Monday*" to specify the departure date, it is better to use the sentence "Do you want to leave *this Monday*?" instead of "Do you want to leave on*February 19th*?" to confirm it.

7.1.5. Confirmation Mechanism Evaluation. For iterative improvement, it is necessary to evaluate the confirmation mechanisms in order to adjust them. The proposed measurements are:

- *For Implicit confirmation*: The recognition rate for each item by comparing the recorded audio files and the word that was recognized as the 1st candidate.
- *For Semi-implicit confirmation*: Number of times the user wants to correct. If this number is high, we should change to explicit confirmation; otherwise, we should use an implicit confirmation strategy.
- *For Explicit confirmation*: Number of times the user denies the system proposal. If this number is low we should consider relaxing the confirmation strategy to an implicit one.

For all confirmations, we can compute the number of questions, the time needed to confirm the item values and the number of times the user rejects the system proposal. To analyze the impact of the confirmation mechanisms on the dialogue speed, the system can compute the percentage of implicit vs. explicit confirmations. The dialogue is faster when the percentage of implicit confirmations is higher. In the analysis of field experiments, we present the final results for the confirmation evaluation (Table 6). In this analysis, we have not included yes/no questions (they are not confirmed because of the high accuracy) nor the rejected answers. When an implicit confirmation is corrected by the user (less than 10% of the time), we have considered it as an explicit confirmation because the cost in time is similar to that of an explicit confirmation.

7.2. General Functionality

Any spoken dialogue system must provide general functionality that permits the user to modify the interaction. These capabilities fit the interaction to the expectations of the user, thus increasing the call success rate.

7.2.1. Error Recovery Mechanisms. When an automated system uses implicit confirmations, it is necessary to define mechanisms to permit the user to recover from system errors. Two commands are available in this process:

START OVER: This command permits the user to start from scratch. Instead of resetting all items, our system begins by confirming groups of items explicitly (dialogue steps). When one group is not confirmed, the system starts from that point:

S: "The selected option is an Intercity train..."
U: "start over"
S: "Let us try again. Do you want to go from Madrid to Barcelona?"
U: "yes"
S: "Do you want to travel on 19th of July?"
U: "No"
S: "Do you want to leave this week, next week or later?"

CORRECT: When the system makes a mistake and takes a wrong item value as right (in an implicit confirmation), the user can correct the system by saying this command at any point of the dialogue. In this case, the system asks again the last item introduced. For example:

S: "On which month do you want to leave?"
U: "May, please"
S: "Which day of March?"
U: "Correct"
S: "On which month do you want to leave?"
U: "May"

7.2.2. General Mechanisms. Other general mechanisms consist of the use of the following commands:

- **HELP**: At any point in the dialogue the user should be able to ask for information. In this case, the system must provide context information depending on the dialogue point. This help should include a comment on the possible problems detected by the speech system (e.g., noise, speaking far from the phone set and speaking lower), and the user response characteristics that the system expects. For the second aspect, it is very useful to include an example (e.g., "the system needs you to specify a departure day, for example 'today'").
- **REPEAT**: In Spoken Dialogue Systems over the telephone, the user can lose his/her attention to the system for several reasons (because of an interruption, for example). In these cases, the user gets confused and he/she does not know how to continue, thus terminating the interaction. Because of this, it is necessary to include the repeat option, permitting the user to ask for repetition when he/she has missed part of the information.
- **STOP AND HOLD**: When the interruption takes several seconds, the repeat function is not enough. In this case, the user needs to stop the interaction and hold it at a specific dialogue point. With this possibility, the user does not have to finish the interaction and call again later (beginning again from the scratch). When the user wants to continue, he/she has to pronounce a specific command for 'waking up' the system.

  When designing the commands/expressions for activating this function, some aspects must be kept in mind:

  o The selected commands/expressions must be phonetically different from the words contained in the recognition vocabularies, but they must be very user friendly.
  o The sentence used to report on the recognition of one of these commands, has to be carefully designed. It has to permit a smooth transition in the dialogue.
  o It is necessary to specify the dialogue points where the system should inform the user on these possibilities: e.g., in the initial help and in the help provided when the system detects problems in the interaction.
  o When one of these commands is recognized, the confirmation strategies have to be more pessimistic about the confidence. One recognition error in these commands, can produce important delays in the interaction.

- **BARGE IN**: this function permits the user to interrupt the system prompts, making the dialogue faster.

When the user knows the item the system is prompting, he/she can interrupt it providing the item value required. In this case, the system stops prompting and starts recognizing the user answer.

7.3. User-Modeling

The user modeling technique applied is based on using four user skill levels (Veldhuijzen van Zanten, 1999). Depending on the current level, the prompt sentences are clearer (they contain more information on how the user should answer, the allowed values, etc.) or the system provides more or less information per time unit. The current skill level changes throughout the dialogue, depending on the positive and negative events occurring during the interaction (Hirschberg et al., 2001).

The user-modeling technique proposed in this paper needs to define three points: the modifiable aspects of the dialogue, the positive and negative events associated with each modifiable aspect (user skill indicators), and the event combination for defining the level change criteria.

7.3.1. Modifiable Dialogue Aspects. These aspects are changeable when the system modifies the current skill level. In this section, we describe some aspects but it does not mean that all of them should be considered as modifiable in all systems:

- *System prompts*: Depending on the user's skill, the prompt could be brief (expert user) or should include a complete help assistant on how to interact with automatic systems (new or inexperienced user). The prompt adaptation is very important to guarantee that the user response is included in the system capabilities (Jokinen et al., 2001). In this aspect, we have defined 4 skill levels:
  - **1st level.** The prompt explains *how to interact* with the system, the *asked item*, the possible *accepted values* and *how to specify* one of the them (e.g., for the period of the day: "Please *speak after the tone*. Say the *period of the day* you want to travel in, in the morning, in the afternoon or in the evening.").
  - **2nd level.** The prompt includes the *item* needed, the *accepted values* for it and the *way to specify them*. ("Say the *period of the day* you want to travel in, *in the morning, in the afternoon or in the evening*.")
  - **3rd level.** Only the required item is included in the prompt. ("Say the *period of the day* you want to travel in.")
  - **4th level.** The user knows everything (the accepted values and the way to specify one of them), and we can relax the question ("When do you want to leave?").
- *System Help*: Similar to the above comment the help should also depend on the user's skill. Related to this aspect, we have considered three different levels:
  - **1st level.** The help provides information on the *general causes of recognition errors* in spoken dialogue systems (e.g., noise, lower speech), the *main limitations of the service* provided by the system (e.g., the system does not provide information about bargains), and *how to answer* the system questions (e.g., providing an example).
  - **2nd level.** In this level, the system eliminates the information on the general causes of error and keeps *the service limitations* and the information on *how to answer*.
  - **3rd level.** In the third level, the system explains only *how to answer* the system questions.

  Gorrel (2002) proposes a complex model for designing different help levels.
- *Confirmation mechanisms*: When the users present a high degree of skill during the interaction (the user knows the system's limitations and the error recovery strategies), the system could make more risky confirmations in order to increase the dialogue speed. On the other hand, when the user does not know how to interact with the system very well, the confirmation mechanisms should be more conservative. The system could modify the confirmation mechanisms, defining different confirmation strategies for the same confidence areas or changing the thresholds that define them (Fig. 8).

  Defining different confirmation mechanisms for the skill levels requires the correct word and error distributions in each level to be represented (as a function of the confidence value). This representation can be obtained from test user evaluations or can be estimated during the real service. Once the correct word and error distributions have been calculated, the confirmation mechanisms can be designed automatically by considering the ideas shown in Section 7.1.3.
- *Subdialogue design*: The sub-dialogues can have a different number of interactions depending on

the questions designed for item prompting. With advanced users, it is possible to ask several items in the same question, thus reducing the sub-dialogue length.

7.3.2. User Skill Indicators. These indicators are defined as dialogue events that provide a hint about the user skill in the interaction with the system. These events can be negative (lack of skill) or positive (high level of skill). Several examples of these indicators are outlined below:

- Positive skill indicators:
  - The number of times the user confirms the system proposal.
  - The number of times the user is able to correct a system error.
  - A high percentage of implicit confirmations.
- Negative skill indicators:
  - The number of times the user does not confirm the system proposal.
  - The number of times the user cannot correct a system error.
  - A high percentage of explicit confirmations.
  - The number of times the user does not respond (remains silence).
  - The number of times the user starts from scratch or asks for repetition.

7.3.3. Events Combination for Changing Level Decision. Finally, we need to associate each modifiable aspect with several positive and negative events. For example, we can associate the system prompts with negative events like the number of times the user keeps in silent or does not confirm the system proposal, and positive events like user confirmations and corrections. Secondly, we combine all the events (positive and negative separately) in order to obtain a single positive and negative measurement. In this proposal, we present a simple method based on a linear combination (Eqs. (1) and (2)):

$$\text{Positive Measure} = \sum_{i=0}^{N} \text{Positive Event}_\text{i} \times \text{Weight}_i \tag{1}$$

$$\text{Negative Measure} = \sum_{i=0}^{N} \text{Negative Event}_\text{i} \times \text{Weight}_i \tag{2}$$

Once single measurements are defined, we should establish the criteria for changing the skill level depending on the evolution of these measurements. When the positive or negative measurement reaches an accumulative value (threshold), the system increases or decreases the level, respectively. The thresholds can depend on the current skill level. The thresholds and the combination weights can be defined by the designer or learned from dialogue transcriptions. Positive and negative measurements are accumulative values but when several negative events occur sequentially, making the negative measurement reaches a threshold, the skill level changes and both measurements are reset. Same for positive events.

In our system, we have considered only the prompts as a modifiable aspect of the dialogue. For the calculation of the positive and negative measurements, we have considered all the weights equal to 1. In our case, the positive and negative measurements are simply the sum of positive and negative events that have occurred sequentially. The evolution of the skill level is represented in Fig. 9. The system starts at the 2nd level (after providing a general explanation of how to interact with the system). When several negative (or positive) events occur, the system decreases (or increases) the level, adapting the interaction dynamically to the user's skill (e.g., asking more explicit questions when recognition errors occur). The number of positive or negative events that forces a change depends on the current level.

Below is an example of a dialogue dynamically modified based on the user's perceived skill level:

[The system is set at level 3]
S: "Say the period of the day you want to travel in."
U: "After lunch"
[The system recognizes "in the evening"]
S: "Did you say in the evening?"
U: "No"
[The system decreases the level from 3 to 2]

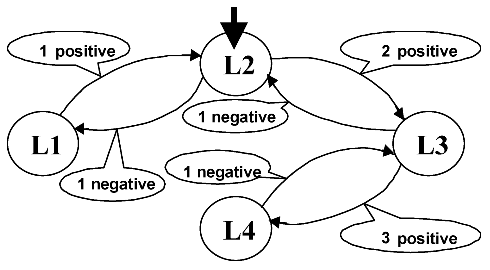


Figure 9. Diagram of transitions between levels in the user-modeling technique.

Table 6. Measurements calculated by the system.

Measure	Global	University	Employers
Call duration (seconds)	195	205	185
Number of questions per call	18.8	19.4	18.3
Average skill level (user-model)	1.88	1.90	1.86
Skill level variance (user-model)	0.32	0.34	0.30
% of implicit confirmations	71.0	74.0	68.0
% of explicit confirmations	29.0	26.0	32.0
Number of START OVER commands per call	0.08	0.14	0.02
Number of CORRECT commands per call	0.43	0.64	0.23
Duration of Negotiation (seconds)	58	57	59
Number of REPEAT commands in the negotiation per call	0.18	0.16	0.20

S: "Say the period of the day you want to travel in; in the morning, in the afternoon or in the evening."
U: "In the afternoon"

In this example, we can see how a more complete prompt sentence helps the user to provide one of the expected inputs.

When the user creates several scenarios in the same call, the system could start the new scenario one level higher than the level at the end of the previous one. This is possible because when the user has completed an scenario it means that the user has answered the system questions once successfully. For simplicity, we have not made the confirmation strategies depend on the user's degree of ability but it would be possible to update the confirmation mechanism at each level. Considering this user-modeling technique, one interesting measure for evaluating the adaptability of the system is the average and variance of the skill level during the whole dialogue (see Table 6). Higher variance reveals a greater adaptability and flexibility of the user-modeling technique.

8. Evaluation of the System

In this evaluation, 105 users called the system to complete four scenarios. There were two types of user: 50 people from the university (students, staff and professors) and 55 users were employers of the Spanish Railway Company, RENFE. The system received a total of 335 calls (not all the users completed all the scenarios). In the information provided to the user, we suggested four scenarios but we also invited the users to include new different scenarios. At the end of the call, each user completed in a questionnaire. 19.5% of the calls asked information concerning the two legs in a round-trip, and 21.3% of the calls completed the reservation for single or round-trips. In 25.1% of the calls, the user hung up before finishing the interaction, so 74.9% of the calls were successful. The evaluation measurements obtained during the test came from the system annotations (Table 6) and from user answers to the questionnaire (Table 7). In these tables, we represent the global average measurements (Global), the university people average (University) and the RENFE employers average (Employers).

As we can see, the average call duration is 195 seconds, greater than that of the operator-based service (152 seconds), but similar to other automated services (Baggia, 2000). The train options negotiation takes around one minute (58 seconds). Regarding the recognition rates, we achieved more than 95% for small vocabularies (less than 50 words/expressions: weekdays, period of the day, etc.). For departure and arrival cities (770-word vocabulary), we obtained an 87.2% recognition rate for in-vocabulary cities and non-rejected answers. 25% of the incorrect cases were solved with

Table 7. Measurements (out of 5) obtained from the questionnaire.

Measure	Global	University	Employers
User experience in this kind of system.	2.8	2.9	2.7
The system understands what I say.	3.1	3.8	2.3
The system responses are clear and concise.	3.4	4.3	2.5
I understand what the system says.	3.5	4.5	2.5
I get railway information fast.	2.9	3.6	2.2
The system is easy to use.	3.5	4.2	2.8
The system helps me during the interaction.	3.1	3.9	2.2
The system is easy to learn.	3.7	4.3	3.1
In case of error the correction was easy.	2.9	3.5	2.2
The system asks me in a logical order.	3.6	4.3	2.8
Generally, it is a good system.	3.0	3.9	2.1

the second candidate and 36% with the spelled-name recognizer. For the remaining cases, more interactions were necessary. In these experiments, we got 3.4% out-of-vocabulary cities, detecting 32% of the cases with the spelled-name recognizer. For the remaining 68% of the cases, the user hung up after several trials.

Because of the high recognition rate and the confidence measurements, the percentage of implicit confirmation was relevant (71.0%). These confirmations made the dialogue faster. In this analysis, we have not included the yes/no questions (they are not confirmed because of their high degree of accuracy) or the rejected answers. When an implicit confirmation is corrected by the user (less than 10% of the time), we have considered it as an explicit confirmation because the cost in time is similar to that of an explicit confirmation.

In Table 7, we present the results obtained from the questionnaire. The users evaluated several subjective aspects on a 5-point scale. The average user experience in this kind of system was 2.8 (evaluated on a 5-point scale: very low, low, medium, high, very high). This data reveals that user experience with spoken dialogue systems is not very high. Generally, most of the aspects obtained a score higher than three (the average value in a 5-level scale: fully disagree, disagree, neutral, agree, fully agree). The worst aspects were those related to the flexibility of the system: error correction and system agility. The dialogue point with the most problems was the departure date specification, because we used isolated speech recognition, and several interactions were required to get a date. The system intelligibility obtained a score of 3.5 (out of 5) due to the use of our restricted-domain female-voice synthesis (Montero, 2000). The dialogue flow obtained the best score (3.6) because of the detailed analysis carried out. This fact makes the system very easy to learn (3.7). At the end of the questionnaire, we asked the users for theirs preference when obtaining train information. 35.2% of the people preferred the system, 35.2% preferred web access and 25.7% preferred to go to the ticket office (3.9% insisted on human operators, although it was not a valid option).

Considering each type of user separately, the behavior has been quite different: the users from the university were more patient with the system, obtaining a good percentage of successful calls (92.7%), although the average call time increased to 205 seconds. They gave the system a 3.9 score in a 5-point scale. The RENFE employers obtained lower percentage of successful calls (57.8%) and lower average call duration (185 seconds). The RENFE employers were more critical of the system, they averaged a 2.1 score on a 5-point scale even though the system solved more than half of the calls within an average call duration of three minutes. The reason of this difference is that part of the RENFE employers work providing train information so they perceive the system as a threat for their employments. In a real case, the users will not be as critical as the RENFE employers but also not as comprehensive with the new technology as people from the university.

9. Conclusion

In this work, we propose a new methodology for designing dialogue managers in automatic telephone-based spoken services. This methodology has been applied successfully to a train information system in Spanish. A combination of several sources of information is proposed: intuition, observation and simulation for defining and evaluating several dialogue strategies and choosing the best one.

The first steps are a database analysis (E-R diagram) and design by intuition, where “brain-storming” on the E-R is carried out for proposing different dialogue alternatives and for defining an evaluation table. In design by observation, we evaluate each proposal using user-operator dialogue transcriptions, without having any system implemented. The limitation to the observation step is that human-human interactions are different from human-system ones. This problem is solved by the Wizard of Oz tool, which simulates the human-system interaction. In the simulation step, we analyze the Wizard of Oz tool and we propose several evaluation measures for all the aspects related to dialogue design. In design by iterative improvement, we describe an approach to incorporate recognition confidence measurements in the definition and management of the confirmation mechanisms. For all vocabularies, this approach facilitated a recognition rate higher than 87%. Two mechanisms for error recover are described: Start-Over and Correct. Finally, a user-modeling technique is described and incorporated into the system for adapting the system dynamically to the user’s ability. At any step in the methodology, new dialogue characteristics can appear, making it necessary to re-define and re-evaluate some aspects of the previous steps, partly repeating the process. Because of this, the methodology must be interpreted as a 5-step iterative process instead of a rigid sequential one.

In this methodology, the observation step is not always possible because most of the times there is not a user-operator system with the same characteristics. In this case, we can skip it and pass to the design by simulation, where we must evaluate the dialogue alternatives.

With this proposed methodology, we have implemented a fully automated system with good user acceptability. In the evaluation, 74.9% of the calls were successful and the average duration for all calls (successful and non-successful) was 195 seconds, similar to Baggia (2000). The users validated the applicability and usability of the system, giving a general score of 3.0 (out of 5).

In this paper, we also provide a review of the dialogue models considered in the international community, plus an analysis of the evaluation problem in spoken language systems.

Acknowledgments

This work has been partially supported by the grant 2FD1997-1062-C02 (EU and Spanish CICYT). The authors want to thank the contributions of colleagues at GTH and RENFE railway company. They also want to thank the reviewers and the editor (Daryle Gardner-Bonneau) theirs comments to improve the paper.

Note

1. If there are several railway stations in the same city, the system should consider all the railway stations during the search for the best train travel.

References

Aust, H., Oerder, M., Seide, F., and Steinbiss, V. (1995). The Philips automatic train timetable information system. *Speech Communication, 17*:249–262.

Baggia, P., Castagneri, G., and Danieli, M. (2000). Field trails of the Italian ARISE train timetable system. *Speech Communication, 31*:356–368.

Bennacef et al. (1996). Dialog in the RAILTEL telephone-based system. *Proceedings of International Conference on Spoken Language Processing*, Philadelphia, USA, pp. 550–553.

Bernsen, N.O., Dykjaer, H., and Daykjaer, L. (1998). *Designing Interactive Speech Systems. From First Ideas to User Testing*, Springer Verlag.

Carpenter, P., Jin,C., Wilson D., Zhang, R., Bohus, D., and Rudnicky, A. (2001). Is this conversation on track? *Proceeding of EUROSPEECH.* Aalborg, Dinamarca, vol. III, pp. 2121–2125.

Córdoba, R., San-Segundo, R., Montero, J.M., Colás. J., Ferreiros, J., Macías-Guarasa, J., and Pardo, J.M. (2001). An interactive directory assistance service for spanish with large vocabulary recognition. *Proceedings of EUROSPEECH.* Aalborg, Denmark, vol. II, pp. 1279–1282.

Constantinides, P.C., Hansma, S., Tchou, C., and Rudnicky, A. (1998). A schema based approach to dialog control. *Proceeding of International Conference on Spoken Language Processing*. Sidney, Australia.

Doran, C., Aberdeen, J., Damianos, L., and Hirschman, L. (2001). Comparing several aspects of human—computer and human—human dialogues. *SIGDIAL 2001 WORKSHOP*, Sept. 1–2, Aalborg, Denmark, pp. 48–57.

Gao, Y., Erdogan, H., Li, Y., Goel, V., and Picheny, M. (2001). Recent advances in speech recognition system for IBM DARPA. *Proceedings of EUROSPEECH*, Aalborg, Denmark, pp. 503–507.

Goddeau, D., Meng, H., Polifroni, J., Seneff, S., and Busayapongchai, S. (1996). A form-based dialogue manager for spoken language applications. *Proceeding of International Conference on Spoken Language Processing*, Philadelphia, USA.

Gorrel, G., Lewin, I., and Rayner, M. (2002). Adding Intelligent help to mixed-initiative spoken dialogue systems. *Proceeding of International Conference on Spoken Language Processing*. Denver Colorado, USA, pp. 2065–2069.

Hacioglu, K. and Ward, W. (2002). A figure of merit for the analysis of spoken dialog systems. *Proceeding of International Conference on Spoken Language Processing*. Denver Colorado, USA, September, pp. 877–880.

Hirschberg, J., Swerts, M., and Litman, D. (2001). Labeling Corrections and Aware Sites in Spoken Dialogue Systems. *SIGDIAL 2001 WORKSHOP*, Sept. 1–2, Aalborg Denmark, pp. 72–79.

Jokinen, K. and Wilcock, G. (2001). Confidence-based adaptivity in response generation for a spoken dialogue system. *SIGDIAL 2001 WORKSHOP*, Sept. 1–2. Aalborg, Denmark, pp. 80–89.

Lamel, L.F., Bennacef, S.K., Rosset, S., Devillers, L., Foukia, S., Gangolf, J.J., and Gauvain, J.L. (1997). The LIMSI RailTel system: Field trial of a telephone service for rail travel information. *Speech Communication, 23*:67–82.

Lamel, L., Rosset, S., Gauvain, J.L., Bennacef, S., Garnier-Rizet, H., and Prouts, B. (2000). The LIMSI ARISE system. *Speech Communication, 31*:339–355.

Lavelle, A.C., Calmes, H., and Pérennov, G. (1999). Confirmation strategies to improve correction rates in a telephonic inquiry dialogue system. *Proceedings of EUROSPEECH*, Budapest, Hungary, vol. 3, pp. 1399–1402.

Litman, D. and Pan, S. (2002). Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction, 12*(2/3):111–137.

Macías-Guarasa, J., Ferreiros, J., Colás, J., Gallardo, A., and Pardo. J.M. (2000a). Improved variable list preselection list length estimation using NNs in a large vocabulary telephone speech recognition system. *Proceeding of International Conference on Spoken Language Processing, II*, pp. 823–826. Beijing, China.

Macías-Guarasa, J., Ferreiros, J., San-Segundo, R., Montero, J.M., and Pardo, JM. (2000b). Acoustical and lexical based confidence measures for a very large vocabulary telephone speech hypothesis-verification system. *Proceeding of International Conference on Spoken Language Processing*. Beijing, China, vol. IV, pp. 446–449.

Minker, W. (2002). Overview of recent activities in speech understanding and dialogue systems evaluation. *Proceeding of International Conference on Spoken Language Processing*. Denver Colorado, USA, Sept., pp. 265–269.

Montero, J.M., Córdoba, R., Vallejo, J.A., Gutiérrez-Arriola, J., Enríquez, E., and Pardo, J.M. (2000). Restricted-domain female-voice synthesis in Spanish: From database design to a prosodic modeling. *Proceeding of International Conference on Spoken Language Processing*, Beijing, China.

Padrell, J. and Hernando, J. (2002). ACIMET: Access to meteorological information by telephone. *Proceeding of International Conference on Spoken Language Processing*. Denver Colorado, USA, Sept., pp. 2713–2717.

Pellom, B., Ward, W., and Pradhan, S. (2000). The CU communicator: An architecture for dialogue systems. *Proceeding of International Conference on Spoken Language Processing*, Beijing, China. vol. II, pp. 723–726.

Riccardi, G. and Gorin, A. (2000). Stochastic language adaptation over time and state in natural spoken dialog systems. *IEEE Transactions. on Speech and Audio Processing, 8*:3–10.

Rudnicky, A., Bennet, C., Black, A., Chotomongcol, A., Lenzo, K., Oh, A., and Singh, R. (2000). Task and domain specific modelling in the Carnegie Mellon Communicator system. *Proceeding of International Conference on Spoken Language Processing*, Beijing, China, vol. II, pp. 130–133.

San-Segundo, R., Montero, J.M., Colás, J., Gutiérrez, J., Ramos, J.M., and Pardo, J.M. (2001a). Methodology for dialogue design in telephone-based spoken dialogue systems: A Spanish train information system. *Proceedings of EUROSPEECH*, Sept. 3–7, Aalborg, Denmark, vol. III, pp. 2165–2169.

San-Segundo, R., Montero, J.M., Gutiérrez, J., Gallardo, A., Romeral, J.D., and Pardo, J.M. (2001b). A telephone-based railway information system for Spanish: Development of a methodology for spoken dialogue design. *SIGDIAL 2001 WORKSHOP*, Sept. 1–2, Aalborg, Denmark, pp. 140–149.

San-Segundo, R., Montero, J.M., Ferreiros. J., Córdoba, R., and Pardo, J.M. (2001c). Designing confirmation mechanisms and error recover techniques in a railway information system for Spanish. *SIGDIAL 2001 WORKSHOP*, Sept. 1–2, Aalborg, Denmark, pp. 136–139.

San-Segundo, R., Macías-Guarasa, J., Ferreiros, J., Martín, P., and Pardo, J.M. (2001). Detection of recognition errors and out of the spelling dictionary names in a spelled name recognizer for Spanish. *Proceedings of EUROSPEECH*, Sept. 3–7. Aalborg, Denmark, vol. IV, pp. 2553–2557.

San-Segundo, R., Colás, J., Córdoba, R., and Pardo, J.M. (2002). Spanish recognizer of continuously spelled names over the telephone. *Speech Communication, 38*:287–303.

Schrâmm, H., Rueber, B., and Kellner, A. (2000). Strategies for name recognition in automatic directory assistance systems. *Speech Communication, 31*(4) 329–338.

Sturm, J., den Os, E., and Boves, L. (1999). Issues in spoken dialogue systems: Experiences with the Dutch ARISE System. *Proceedings of ESCA Workshop on Interactive Dialogue in Multimodal Systems*. Kloter Irsee, Germany, pp. 1–4.

Suhm, B. and Peterson, P. (2002). A data-driven methodology for evaluating and optimizing call center IVRs. *International Journal of Speech Technology, 5*:23–37.

Sutton, S., Cole, R.A., de Villiers, J., Schalkwyk, J., Vermeulen, P., Macon, M., Yan, Y., Kaiser, E., Rundle, B., Shobaki, K., Hosom, J.P., Kain, A., Wouters, J., Massaro, M., and Cohen, M. (1998). Universal speech tools: The CSLU toolkit. *Proceeding of International Conference on Spoken Language Processing*, Sydney Australia, pp. 3221–3224.

Trías Sanz, R. and Mariño, J. (2002). Bassurde[lite], a machine-driven dialogue system for accessing railway timetable information. *Proceeding of International Conference on Spoken Language Processing*. Denver Colorado, USA, pp. 2685–2689.

Veldhuijzen van Zanten, G., (1999). User modelling in adaptive dialogue management. *Proceedings of EUROSPEECH*, Budapest, Hungary, vol. 3, pp. 1183–1186.

Walker, M.A., Litman, D.J., Kamm, C.A., and Abella, A. (1997). PARADISE: A general framework for evaluating spoken dialogue agents. *ACL/EACL*, 271–280.

Walker, M.A., Litman, D.J., Kamm, C.A. and Abella, A. (1998). Evaluating spoken dialogue agents with paradise: Two case studies. *Computer Speech and Language, 12*:317–347.

Walker, M. (2001). Darpa Communicator dialog travel planning systems: The June 2000 data collection. *Proceedings of EUROSPEECH,* Aalborg, Denmark, pp. 1371–1374.

Walker, M., Rudnicky, A., Prasad, R., Aberdeen, J., Owen, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Roukos, S., Sanders, G., Seneff, S., and Stallard, D. (2002a). Darpa Communicator: Cross-system results for the 2001 evaluation *Proceeding of International Conference on Spoken Language Processing*., Denver Colorado, USA, pp. 269–273.

Walker, M., Rudnicky, A., Prasad, R., Aberdeen, J., Owen, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Roukos, S., Sanders, G., Seneff, S., and Stallard, D. (2002b). Darpa communicator evaluation: Progress from 2000 to 2001. *Proceeding of International Conference on Spoken Language Processing*, Denver, Colorado, USA, pp. 273–277.

Ward, W. and Pellom, B. (1999). The CU communicator system. *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding* (*ASRU*), Keystone Colorado.

Zhang, J., Ward, W., Pellom, B., Yu, X., and Hacioglu, K. (2001). Improvements in audio processing and language modeling in the CU communicator. *Proceedings of EUROSPEECH*. Aalborg, Denmark, pp. 2209–2212.

Zue, V., Seneff, S., Glass, J., Hetherington, L., Hurley, E., Meng, H., Pao, C., Polifroni, J., Schloming, R., and Schmid, P. (1997). From interface to content: Translingual access and delivery of on-line information. *Proceedings of EUROSPEECH*, Athens, Greece, pp. 2227–2230.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T.H., and Hetherington, L. (2000). JUPITER: A telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing, 8*(1):85–95.