Philips J. Res. **49** *(1995) 399–418*

A SPOKEN LANGUAGE INQUIRY SYSTEM FOR AUTOMATIC TRAIN TIMETABLE INFORMATION

by HARALD AUST, MARTIN OERDER, FRANK SEIDE and VOLKER STEINBISS*

Philips GmbH Forschungslaboratorien, Postfach 1980, D-52021 Aachen, Germany

Abstract

This article describes the Philips automatic train timetable information system which enables the user to call up accurate information about train connections between 1200 German cities over the telephone. In contrast to most of the inquiry systems available so far, the caller can talk to our system in unrestricted, natural and fluent speech, very much like talking to a human operator. No instructions are given beforehand.

The system consists of four main components: speech recognition, speech understanding, dialogue control, and speech output. They are separated into independent modules and executed sequentially. The speech recogniser creates a word graph from the spoken input. This word graph is then passed to the understanding component which computes the meaning, using an attributed stochastic context-free grammar. A dialogue manager analyses the results and either accesses the database or comes up with another question if necessary.

The system has been made available to the general public in an ongoing field test, both to gather speech data and to evaluate its performance.

Keywords: automatic inquiry system; speech recognition; speech understanding; natural language understanding; word graph; dialogue control; train timetable information.

1. Introduction

Automatic inquiry systems are systems that people can call in order to obtain certain information, without a service representative being involved. In order to create a database query from the spoken input, the system requires

*) This work was in part supported by the German Ministry of Research and Technology (BMFT) under contract No. 01 IV 102 B.

not only speech recognition, but also a speech understanding component to determine the meaning of the spoken input, and components for dialogue control and speech output.

Most of the automatic inquiry systems that are already in operation are not very user-friendly. Callers have to interact with them either by pushing keys on their touch-tone telephone, or by uttering one of just a few words the system can understand. This leads to rigidly structured, usually menu-driven dialogues that are accompanied by lengthy explanations.

In our system, in contrast, the user can talk in unrestricted natural and fluent speech, very much like he would chat with a human operator. He does not receive any instructions beforehand.

Our prototype installation provides information on the schedule of the German railway system. It understands and responds in German; all examples in this article were translated into English. The vocabulary of the speech recogniser currently consists of approximately 1850 words, 1200 of which are railway station names. They are made up of the most important German cities, plus a number of smaller stations in the Aachen area and a few from abroad. Even though there are almost 10000 railway stations in Germany, the above described selection ensures that more than 95% of the timetable inquiries occurring in real life are covered.

The current system is a pure software solution (written in C) that runs on Unix workstations. Therefore, response times may depend on the current load of the computers, but we usually attain real-time operation. The system is connected to the public telephone network and can be reached under the number +49 241 604020 from all over the world. It should be made clear at this point that our system is still a research prototype that is in no way interrelated with the official railway information service of the Deutsche Bahn AG. The database we use, though, is a real one, so the information a caller gets is accurate.

A typical call to the system goes as follows:

System: Good morning. This is the automatic timetable information service. From where to where do you want to go?
Caller: Good morning. I need a connection from Munich to Hamburg.
System: When would you like to go from Munich to Hamburg?
Caller: Next Saturday, er, at seven o'clock.
System: So you want to travel on Sunday the 19th of February at 7 am?
Caller: No, on Saturday.
System: So you want to travel on Saturday the 18th of February at 7 am?
Caller: Exactly.
System: There is the following connection: With IntercityExpress 682,

departure from Munich at 7:20, arrival in Hamburg at 12:51. Would you like me to repeat the connection?
Caller: No, but I need a train for the return trip on Sunday.
System: At what time would you like to go from Hamburg to Munich on Sunday the 19th February?
Caller: I have to arrive in Munich late in the evening.
System: So you want to arrive in Munich between 8 pm and 11 pm?
Caller: Yes.
System: There are four connections. This is the first one: With IntercityExpress 789, departure from Hamburg at 14:02, arrival in Munich at 20:03. Would you like me to repeat the connection?
Caller: No, thank you.
System: Would you like the next connection?
Caller: No, thanks, this one is fine.
System: Do you need another connection?
Caller: No.
System: I wish you a pleasant journey. Thank you for calling. Good bye.

Just as in the communication process between humans, recognition errors and misunderstandings may occur. The system therefore allows the caller to correct these mistakes, as shown in the example. Under adverse circumstances, it may still happen that a customer cannot reasonably be understood. This will eventually be detected, and the caller will be referred to a human operator.

In Section 2 of this paper, we will introduce the system architecture and describe the individual modules and their functionality. Section 3 reports on our experience with the on-line operation of the system. In Section 4, finally, some conclusions are summarized.

2. System architecture

Most traditional applications of automatic speech recognition like dictation or voice control only perform a speech-to-text conversion. An inquiry system, however, additionally requires speech understanding, dialogue control, and speech output capabilities. The interaction between these basic components is of particular interest, since an appropriate architecture should be expected to help increase the system's overall performance. This is especially important for the integration of the recognition and understanding parts.

The recognition performance is deteriorated by several effects:

- The quality of the incoming speech signal is poor because of the limited-bandwidth and often noisy telephone lines.

- The vocabulary may comprise several thousand words, yet the system has to operate at least close to real-time.
- An automatic inquiry system must, of course, be speaker-independent.

These rather adverse circumstances lead to a word error rate of about 20% in the current system. One could hope to improve this by exploiting the knowledge of the understanding component for recognition purposes, effectively using it as a language model.

In our system, however, we do not integrate the different components directly, but separate them into individual modules that are executed sequentially (Fig. 1). Well-defined interfaces ensure that the system can be easily maintained and that entire modules can be exchanged without affecting the remaining parts. In this way, we avoid complicated and error-prone interface protocols inherent in more complex architectures.

2.1. Speech recognition

Our inquiry system employs the PHICOS system of Philips Research for speech recognition. It uses hidden Markov models with continuous mixture densities, 6-state left-to-right phoneme models, and a tree-organised beam search [1]. The system is described in reference [2]. The vocabulary of the inquiry system consists of a total of 1850 words, 1200 of which are railway station names.

2.1.1. Acoustic modelling

To adjust it to the new application, the PHICOS recogniser had to be modified in several respects. First of all, the feature extraction had to be adapted to the telephone-line bandwidth. We use a mel-scale log filter bank with 14 channels

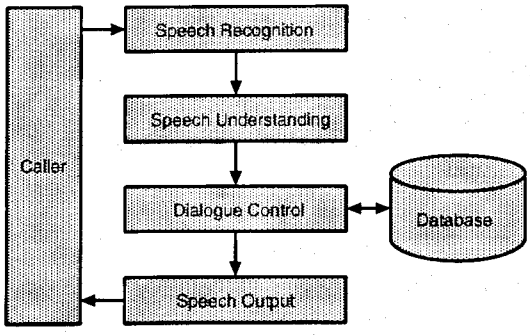

Fig. 1. The system architecture.

in the range from 300 to 3400 Hz. The short-time channel energies are normalized and augmented by their first-order time differences. The overall short-time signal energy and its first- and second-order time differences are also included.

Unlike in the dictation product, we use only monophones. For our task, the reduction of the error rate achieved by employing context-dependent phonetic units did not justify the inevitable increase in computational effort.

The system had to be trained for speaker-independent operation over the telephone network. While the lack of appropriate data was a problem in the beginning, we now have about 20 hours of training material at our disposal.

2.1.2. *Word graphs*

Since the word error rate is still high, as pointed out before, it usually does not suffice to just examine the single best sentence, computed by the recognition. Thus, the meaning of a spoken sentence is computed from *word graphs* that are generated by the recogniser [3]. A word graph (Fig. 2) is a directed acyclic graph whose nodes represent points in time, and whose edges are labelled with a word and an accompanying acoustic score. In our case, the score is the negative logarithm of the joint acoustic probability density. Each path through the graph represents a sentence hypothesis.

The word graphs are generated in a two-step process: First, a *word hypotheses generator* (WHG) produces scored word hypotheses at a rate of several thousands per spoken word. The WHG is basically a time-synchronous Viterbi recognizer that starts a new search tree at every nth frame, where $n=3$ has turned out to be an adequate compromise with respect to accuracy and computational costs. For each path reaching the last state of a word, a word hypothesis is emitted. Using proper pruning techniques, a search effort comparable to the normal integrated bigram search for the best sentence can be obtained.

The second step in word graph generation is the *word graph optimisation*. By optimising over word boundaries, purging and pruning, the number of

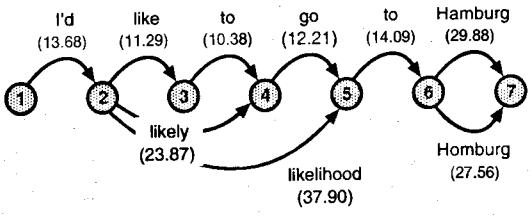

Fig. 2. A sample word graph. The parenthesised numbers are negative logarithms of acoustic probabilities computed by the speech recogniser. Note that the density of word graphs actually occurring in our system is usually much higher.

hypotheses is reduced in order to obtain a relatively sparse word graph. This phase uses the word scores delivered by the WHG and does not need to access the acoustic models any more, and it does not require major computational resources either.

The density of a word graph, measured as the average number of edges created for a spoken word, can be adjusted by tuning the relevant pruning parameters. In our prototype, graphs with about 10 edges per word turned out to be most effective.

2.2. Speech understanding

The word graph that was created by the speech recognition component is then passed on to the understanding module whose task it is to find the best path through the graph, and to determine its meaning.

For our purpose it is not necessary to understand every word in the spoken sentence. Instead, we are only looking for — more or less standardised — phrases that are usually used when someone expresses a certain aspect of a request for a timetable information. These *concepts*, like 'to <station name>' for the destination, may occur in arbitrary order, and they may be interspersed with *filler words* that are meaningless with respect to the creation of the database query, like 'Good morning' or 'I want to go'.

It is thus sufficient to locate and process just the concepts rather than the entire sentence. This technique has two major advantages: sentences that are grammatically incorrect — which is rather common since we are dealing with spontaneous speech — or insufficiently recognised, can still, at least partly, be understood. And the method is computationally inexpensive: on average, our current system needs just 28 ms to understand a sentence.

An *attributed stochastic context-free grammar* serves as a language model, identifies the relevant phrases and computes their meaning.

2.2.1. Bridging silence

In a word graph like the one depicted in Fig. 3, the special word SILENCE can

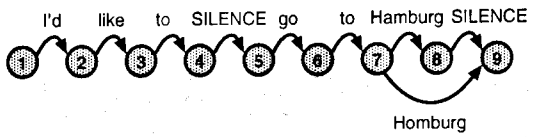

Fig. 3. A very simple word graph that will be used in this article to demonstrate the speech understanding process. The probability scores have been omitted for clarity.

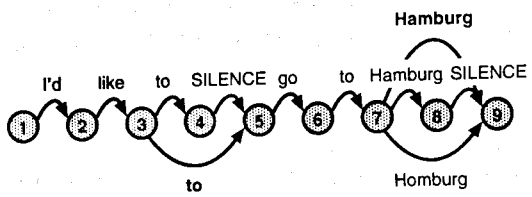

Fig. 4. The modified word graph: SILENCE arcs are bridged.

occur because of a low or hesitating way of speaking. It may also recombine parallel paths into one node.

While SILENCE is a word just like any other for the recogniser, it has to be ignored when searching for meaningful word sequences. Therefore, in a first processing step, for each non-SILENCE arc followed by a SILENCE arc, a new edge is inserted into the graph that extends from the start node of the word arc to the end node of the SILENCE entry (Fig. 4). It is labelled with the same word and the sum of both scores. Processing all nodes in ascending order ensures that sequences of SILENCE arcs are treated adequately as well.

2.2.2. Concept graphs

In the next step, this modified word graph is converted into a *concept graph* (Fig. 5). Its nodes are the same as in the word graph; its edges, however, are *concept instances* instead of words. All entries of the original graph that do not contribute to a concept are missing.

2.2.3. Stochastic grammars for concepts

We use a *stochastic context-free grammar* [4] to model the individual concepts. For each concept, a distinct start symbol is defined, so our grammar can be regarded as a set of grammars that may share subordinate rules. Every rule has a probability which indicates how likely it is to be applied, given the left-hand-side non-terminal. Note that this grammar is semantic rather than syntactic since it does not describe the structure of a sentence but its meaning.

For each of the start symbols, all possible derivations are computed and

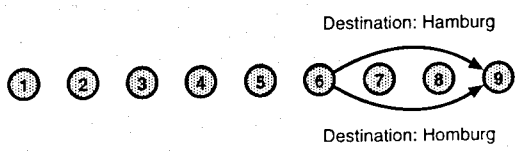

Fig. 5. The concept graph.

inserted into the concept graph. Similar to the word graph, these edges also have a score. Following a standard approach in speech recognition, there are two contributions to this value: the sum of the scores of all word arcs generating the concept instance, corresponding to the acoustic likelihood of this word sequence, and a language model score delivered by the grammar. It is computed by summing up all scores of rules involved in such a partial parse.

The grammar can be augmented by definitions for insignificant but often-used phrases like "I'd like to" or "Could you please". The resulting better coverage of the word graph reduces the risk of erroneously finding concepts that were no element of the utterance.

A concept graph still contains all the information of the original word graph that is relevant with respect to the given grammar. Additional knowledge, in the form of the language model probabilities, was incorporated.

2.2.4. *Filler arcs*

In general, there is no path from the first to the last node of a concept graph, since concepts will not always be adjacent to each other. Besides, recognition errors can cause concept instances to appear that were not part of the spoken sentence. When searching for the best path through the graph, we therefore have to be able to bridge gaps as well as to bypass individual concepts.

For this purpose, *filler arcs* are introduced. They are labelled with an empty concept and the acoustic score of the optimal path between their start and end nodes in the word graph.

For every concept edge, a bypassing filler arc is inserted. Also, begin and end of the concept graph are connected with each concept, and the same is done for all of those concepts that were not interlinked before. The result is a *complete concept graph* (Fig. 6) serving as a suitable basis for determining the most probable path.

2.2.5. *Searching for the optimal path*

However, the best path through a concept graph constructed that way always

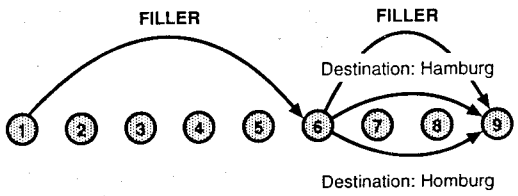

Fig. 6. Finally, the complete concept graph.

consists of filler arcs only. To avoid this, a *filler penalty* is added to the score of each filler arc. It is time-proportional in order to express that sequences of filler words are increasingly unlikely the longer they last.

Since certain concept sequences are observed more frequently than others, we additionally employ a concept bigram model, including sentence beginning and ending. Filler arcs are ignored on this level.

This technique is fairly simple but achieves very satisfying results, provided that the filler penalties are properly chosen. Nonetheless, there is clearly room for improving the modelling.

2.2.6. Determining the meaning

With the method described above, the most likely sequence of concepts (and thus words) can be derived from a given word graph. However, in order to create a database query, we need the meaning of the concepts rather than their textual representation. It is advisable to compute it as soon as the concept graph is constructed. A meaning is associated with each concept instance and not just with those in the chosen path. This allows for a modified path search utilising semantic information later on. This could be formulated as hard constraints ('two destinations being observed in one sentence must be equal'), or once again, as probabilistic constraints ('two different destinations in one sentence are very unlikely').

In order to associate a meaning with the concepts, we extended our grammar to become an *attributed grammar* [4]: each non-terminal can have any number of *attributes*. Every syntactic rule can be provided with *semantic rules*, which compute the attributes of the non-terminal on the left side from the given values on the right; only these so-called *synthesised attributes* are used (Fig. 7). This way, an expression is parsed top-down, and its meaning is subsequently computed bottom-up.

```
<Time>    ::= (0.56) <prep> <Hour> <Minute>
  time    :=  <Hour>.number * 60 + <Minute>.number
<Hour>    ::= (2.61) one | (3.38) two | ...
  number  :=  1 | 2 | ...
<Minute>  ::= (8.31) fifteen | (5.22) twenty | ...
  number  :=  15 | 20 | ...
<prep>    ::= (1.64) at | (2.11) after | ...
```

Fig. 7. A simple stochastic attributed grammar for time expressions. The values in parentheses are proportional to the likelihoods of the corresponding rules.

TABLE I
Sample date representations

Today	`CURRENT_DATE`
Tomorrow	`CURRENT_DATE + 1`
Next Saturday	`$,,,6, += CURRENT_DATE`
Three days before Christmas	`($,12,24,,3 += CURRENT_DATE) - 3`
Within this month	`CURRENT_DATE .. ((,,1,, ++ CURRENT_DATE) - 1)`

`date1 .. date2` results in `date1` with a duration up to the beginning of `date2`.
`date + integer` (`date − integer`) computes the date `integer` days after (before) `date`.

Table I shows a few examples of dates expressed with this formalism.

2.2.8. Disambiguation of concepts

With the process described above, the relevant parts of an utterance and their meaning can be determined. However, individual concepts may be instantiated more than once, or they might be missing altogether. Several reasons can be responsible for that: callers may leave out information or contradict themselves. They may wish to correct themselves, or they may use a phrase not covered by the grammar. Recognition or understanding difficulties can also cause confusion.

Missing information does not constitute a real problem in a dialogue environment — a follow-up question can take care of that — whereas multiple instances are more troublesome. This is because they may be contradictory, but often are not. For example, "at four o'clock" and "in the afternoon" are two distinct time specifications that can, of course, be joined, thereby not only reducing the number of hypotheses, but also rendering unique the previously ambiguous "four o'clock". We use a number of rules for the combination.

2.3. Dialogue control

Once the concepts and their meaning in the spoken sentence are found, it is theoretically possible to create the database query. However, as already mentioned, missing, ambiguous, or contradictory information may prevent just

that. Besides, because of the recogniser's relatively high error rate, it is important that the system can verify what it believes it understood. Otherwise, an incorrect query may result.

It is evident what is to be done to overcome these problems: we need a system that can ask questions and involve the caller in a *dialogue*.

For reasons of user friendliness, it is highly desirable for this dialogue to be flexible, natural and capable of adapting itself to the user's utterances. Unfortunately, this may result in hundreds of different questions that can follow in almost arbitrary order, leading to tens of thousands of possible dialogue situations that cannot be represented in any convenient way.

We have therefore developed a special programming language for dialogues in automatic inquiry systems [6]: most aspects of this subclass of general dialogue, e.g. questions, slot definitions and verification phrases, are specified in a declarative way. An interpreter then takes care of the selection of appropriate questions, handles the results it gets from the speech understanding module, and finally creates the database query. With this description formalism, the complexity of a dialogue representation is considerably reduced.

2.4. Speech output

Both the results from the database query and the questions generated by the dialogue manager are passed to the speech output component in the form of complete, written sentences. Any available speech synthesis program can then be used to convert these words into 'spoken' language.

However, real text-to-speech synthesis as it is available today does not sound completely natural. As the vocabulary is large but fixed in this application, we recorded all necessary phrases and words, e.g. the station names, numbers, and names of months and weekdays, and stored them in digital form on hard disk. Whenever output is to be created, the appropriate segments are concatenated and replayed.

While this approach is fairly simple and somewhat limited in flexibility, the generated speech sounds rather pleasant and seems to be widely accepted.

3. Field test

In order to get training material for the recognition and understanding parts, as well as for evaluation purposes, we had to make people call our system. These realistic calls were also necessary for us to debug the system and learn about customers' demands and ideas, their general behaviour and problems, so we could increase the system's usability and usefulness. And finally, we were interested to see how far such an automatic system would be accepted.

For finding answers to questions like these, *Wizard-of-Oz (WOZ) scenarios* in which a human being takes the part of the computer are widely used [7, 8]. While such a set-up is well suited for the collection of speech data, the other aspects cannot be covered as well. Obviously, the system cannot be tested, debugged and evaluated if replaced by humans during the test. Also, the way people react to a system and their attitude towards it largely depends on its performance. This in turn is closely related to the recognition and understanding rate which is hard to simulate. Therefore, in a WOZ environment, the utterances collected, as well as the observations made and comments given, are less realistic than when the actual system is used.

3.1. Acquiring realistic speech data

For the reasons pointed out above, we conducted a *field test* with our system. The general idea was to tell people about the existence of a new automatic inquiry system and encourage them to use it or at least to try it out. They were not given details or instructions in any case. It should be noted, though, that while our approach allows for a rather exact evaluation, the collected data are not necessarily completely realistic: people who call in because they are curious tend to behave differently from those who really need information.

The field test was organized as a *bootstrapping process*. We began with a speaker-dependent system that was not to be used over the telephone but with an ordinary high-quality microphone, and that displayed its output on the computer screen. Our initial training material for both speech recognition and speech understanding consisted of roughly 1000 sentences that a number of different people had thought up. This data, however, was not very representative for real dialogues. About 15% of the sentences were only slightly related to the conversation subject ("when will the schedule change") and many phrases that are often used in real inquiries were not contained in this data. The best example is the indispensable word "yes" which in the beginning was hardly understood at all since it did not occur in the training material.

With this initial system we were able to collect some speech data that had mostly been spoken by the developers themselves. Having trained the system anew, a telephone version was installed that showed, foreseeably, poor performance. In order to induce other persons to call, the telephone number was first circulated within our research group, and later inside the department and the entire laboratory. After each step, the newly compiled material was harnessed for retraining and general improvements.

Motivating other people to actually call our system turned out to be a major problem. We used personal conversation and posters for advertising. Unlike as described in reference [8], we always pointed out that this was an automatic

system, hoping to raise curiosity in this way. To avoid that only the relatively small number of colleagues actually planning a journey would call, we sought to explicitly encourage everybody to simply try the system and see how it works.

While we received clearly fewer calls than we had expected, those that came in eventually enabled us to improve the system's performance to a point when we were ready to address the general public. Beginning in February 1994, it has been promoted through press releases and radio interviews. The number of incoming calls showed considerable peaks after each publication; often, the system was not idle for more than one or two seconds before the next call arrived. Since we were offering only a single telephone line, we probably missed many calls, in particular because people who call the system only out of curiosity are likely to lose interest if they attempt several times to get through but always hear the busy signal.

Proceeding in the manner described above, we were able to debug, evaluate and improve our system even further. The disadvantage is that many of the incoming calls were made by people who only tested the system and did not really need any information. We would probably have to connect our system to an official information service to overcome this drawback.

3.2. Call transcription

For further training of both the recognition and understanding components, it was necessary to record all incoming calls and transcribe them manually. These transcriptions can also be used to evaluate the dialogue and detect possible problems or awkward passages.

Each call is annotated with attributes that describe its particular properties; voice characteristics (male/female/child), speaker accent, and line quality (low volume/noisy/background talk/background music, etc.) being amongst the most prominent. We also noted whether the call was successful or not, which cannot always be rated because some callers hang up immediately or check out the system instead of seeking a train connection.

The system itself always stores the output it creates, i.e. the questions and the query results, so together with the transcription, the full course of a dialogue can be recovered. It is helpful for the transcriber to see what kind of system action leads to an otherwise inexplicable user response, and in a future version, this information may also be useful for dialogue-situation specific training.

The recogniser's vocabulary is updated automatically. It consists of the words the speech understanding component can process, plus all of those that occurred in callers' utterances at least n times, with n typically being between 2 and 4, depending on the desired vocabulary size.

3.3. Evaluation

While it is trivial to evaluate a stand-alone speech recognizer, and feasible to do so for a system that creates a database query from a single sentence, there is no easy way to evaluate dialogue systems automatically [9]. After all, the all-important criterion is user satisfaction, which cannot be determined as easily as the word error rate, for example.

Probably the best approximation is to measure the percentage of callers who obtained the information they asked for. This, of course, is only a simplified approach since important aspects like the speed of the system, clearness and understandability of the voice output, etc. are not accounted for.

To get at least a rough idea of the callers' opinions, we ask them for general comments and for suggestions for improvements at the end of a successful dialogue. Their remarks are recorded but not processed by the speech recogniser. Detailed questionnaires as used in references [10] or [11] could certainly yield more valuable information, but our experience shows that in an anonymous telephone environment most callers would not bother to answer them. Indeed, only very few people even respond to our short invitation to comment on the system. Those who do, however, are usually enthusiastic; phrases like "very good" or "wonderful" abound. Negative statements mostly refer to difficulties that occurred during the previous dialogue and are hardly ever of a more general kind.

It is interesting to see that evidently many people are not aware that speech recognition is a technology that does not yet work 100% accurately. Problems caused by recognition errors are often not perceived for what they are, but are attributed to general weaknesses of the system. This impression is confirmed by a number of comments explicitly referring to the speech output: it seems that in the minds of the general public speech output, and not speech recognition, is regarded as the real challenge.

Another remarkable observation from the feedback is that satisfaction with and approval of the system are apparently less dependent on the course of the previous dialogue but rather on the general attitude of the caller.

These experiences indicate that a good and human-like speech output is important for a high acceptance of an automatic inquiry system. Hence, our method of using a human voice instead of a synthesised one turns out to be a good choice.

3.4. Observations and problems

In this section, we will give an overview of the observations and problems most frequently noticed during the field test. Interestingly enough, while some of them were foreseeable, others came completely unexpected. This only confirms how valuable a system evaluation in a real-world environment is.

3.4.1. Technical problems

A sizable number of difficulties was caused indirectly by the telephone interface. One problem was that we cannot compensate for different volumes of callers' voices in our system. While some of them spoke so quietly that the system could not find out whether they talked at all, others were so loud that considerable distortions were caused already by their receiver's microphone. Unfortunately, there is little that can be done about that.

Furthermore, our current installation is very simple and does not allow a flexible adaptation to an individual person. We consider a caller's utterance to be finished after a predefined amount of silence went by. If this pause is too long, an awkward delay will result. On the other hand, if it is too short, people speaking hesitatingly may be interrupted in the middle of a sentence.

The detection and processing of the actual speech signal was occasionally also made more difficult by background noise, e.g. from a radio or television set. Some callers even created their own distractions by talking to other people in the room: "Amazing — he understood everything!" (This astonished remark then confused the system completely, and the call was finally aborted.)

3.4.2. Problems with the database

Two problems that we had not anticipated were caused by the database. Firstly, since we create speech output by replaying pre-recorded phrases, everything that is to be converted from written to spoken language must be known in advance. This is especially true for all timetable information returned from the database. We had not expected to encounter words like "ferry" or "footpath" when asking for train connections an consequently could not output them.

Secondly, whenever we received a new release of the database because of the seasonally changing train schedule, we found that some stations did not exist any more or were named differently. The solution here would be a closer cooperation with the manufacturer of the database or generation of synthetic speech.

3.4.3. Recognition errors

While much of the natural language understanding and dialogue research efforts in the past went into systems that accept typed input, our system by definition deals with spoken language. The major difference is that in our case recognition errors can and will occur. In fact, because of the adverse conditions of speaker independence, spontaneous speech, low signal quality, open and relatively large vocabulary and real-time constraints, the word error rate of the recogniser is currently about 20%.

In an inquiry system, recognition errors can have two effects: something said by the caller may not be correctly understood, and something not said might be erroneously found in the speech signal. The first case is not too troublesome if it doesn't occur too often — an appropriate question will cause the caller to repeat what he or she said — whereas the second turns out to be more severe. People tend to be confused if the system 'knows' something they have not yet talked about, and may not be able to correct such a misunderstanding. Therefore, the dialogue should be designed in a way that at any particular time it can only understand those things that a caller can be reasonably expected to say.

The disadvantage of this strategy is that there will always be people who say something that makes sense in certain circumstances, but the system does not comprehend it. On the other hand, a caller cannot distinguish such a situation from a simple recognition or understanding problem, and these are unavoidable, anyway.

A typical example where we changed the dialogue in order to account for the recognition difficulties outlined above is the way we handle verifications. Due to the high error potential, it is vital for an automatic inquiry system to verify what it believes it understood by asking appropriate questions. Otherwise, an incorrect query may result.

Our first approach was to come up with a single verification question like "So you want to go from Hamburg to Munich tomorrow at 3 pm?" after all information was gathered, and to allow the correction of all data at this point. The advantage of this approach was that we did not have to interrupt the normal flow of conversation. Unfortunately, the consequence of this strategy often was that a confirmation given by the caller, like "Yes, exactly", was misrecognized as correction, e.g. "at 4 pm". In some cases, the result was that while the system originally had understood everything perfectly, the unintended correction caused severe problems that finally led to a wrong query or an aborted call.

In our current system, we therefore verify everything by changing the subsequent question appropriately, as in "When would you like to go to Munich?" instead of "When would you like to go?" when the destination "Munich" is to be confirmed. A correction is only possible for those values that appear in the question (in the example, the destination), which greatly reduces the odds of misunderstandings. This strategy has a disadvantage, too: some people do not realize that they can correct a value since they are not explicitly asked to do so.

Another major difference between written and spontaneously spoken input lies in the correctness of the utterances. While many of the observed expressions are grammatically correct sentences or at least parts thereof, word sequences like "twelve midnight for to be in Hamburg" also appear. This

underlines the validity of our understanding approach that only looks for meaningful words and phrases, regardless of their order within an utterance.

3.4.4. Reaction to questions

We found that the way a particular question is asked greatly affects the response of the caller. Our system originally initiated the dialogue with "How can I help you?". This formulation, natural as it may be among humans in a similar situation, apparently confused several people who did not know how to react. When this phrase was altered to a more suggestive "From where to where do you want to go?", quite often, simple answers like "from Hamburg to Munich" were the result.

Of course, there is no guarantee that questions will always be answered in the way one would expect. In fact, we often encountered situations in which a response did not only not answer the question but was, at least at first sight, altogether illogical:

System: When would you like to go to Hamburg?
Caller: No.

We suppose that in this and comparable situations the callers tried to speak in a machine-like language — in this case, to indicate that they did not want to go to Hamburg — because they did not realise that the system would have understood a response like "No, not to Hamburg, I want to go to Munich."

3.4.5. Callers who only test the system

Several callers apparently only tested the capabilities and limitations of our system, and did not need any information or did not even pretend they would. Typical of this is the use of absurd phrases like "I want to go yesterday", "on the 30th of February", or "from Hamburg to Hamburg".

We do not think that these are problems that major effort should be spent on. After all, a realistic caller wants information and can therefore be expected to be cooperative. Besides, it will never be possible to make a system absolutely foolproof anyway.

Another phenomenon was that an unexpectedly high number of stations was asked for that were not in the vocabulary. The explanation is that people who only wanted to try the system often asked for a connection to their home town, or purposely tried small stations to see whether they would be understood. As mentioned above, the selection of the stations in our vocabulary should ensure that more than 95% of real inquiries can be answered.

3.4.6. Accents and dialects

People who speak in German dialect or with strong foreign accents amazingly almost never had problems in being understood. This may partly be due to a particularly careful and emphasized pronunciation typical of non-native speakers.

Occasionally, though, foreigners used phrases that native speakers would never consider and that were not covered by the system's grammar. Consequently, they were not understood, even though it would have been clear to humans what they meant. However, this does not constitute a real problem: if an automatic inquiry system were to be employed in an environment where many non-native speakers could be expected to use it, the appropriate phrases could simply be added to the grammar.

3.4.7. Diverse opinions on details

It was interesting to note that the opinions of people we asked about our system varied, even on very specific topics.

For example, some people found the system's announcements too slow, whereas others complained about their high speed. The comments on the (male) voice of our system range from "very pleasant" to "boring" and "should be changed".

Before our system became fast enough for real-time operation, a piece of (electronic) music was played whenever the caller had to wait for a response. We found that some kind of pause-bridging sound is necessary if a system cannot answer immediately. Otherwise, callers may be confused and wonder whether the line was interrupted. While half of the people asked liked the music, the other half detested it.

The consequence of these observations is that one has to be very careful when it comes to modifications of the system. In particular, it should not be changed because of individual opinions, not even if they are identical to one's own.

3.5. Evaluation results

Currently, we receive about 1500 calls per month; altogether, we have collected more than 10 000 so far. Approximately one-third of them cannot be used for evaluation purposes since they were made by people who only played with the system, used it for party entertainment, or hung up right after the initial greeting. Of the other two-thirds, 10% seem to consist of real requests, while 40% of the callers apparently only try the system. For the remaining 50%, this cannot be decided. The success rate, i.e. the percentage of callers who actually get the information they originally requested, for these three

groups averages about 80%. One quarter of the remaining calls fails due to poor recognition performance, which we hope to improve further as we collect more training data. The rest ask for stations that are not in the vocabulary, or have other problems with the dialogue.

4. Conclusion

We have described the system architecture and the components of our automatic train timetable information system. We also reported on the experiences we gained during the operation of the system in a broad-based and ongoing field trial. This test was organized as a bootstrapping process, which allowed us to start with only little original training data and utilize the incoming calls for improvements from the very beginning. We found that this is a good way to collect training material and to evaluate the system at the same time.

The success rates achieved in the test make clear that the underlying technology is well-suited for realistic applications. Most importantly, the reactions of many callers show that automatic systems of this kind are accepted and welcome since they can often provide better and more easily accessible service.

REFERENCES

[1] H. Ney, R. Haeb-Umbach, B.-H. Tran and M. Oerder, Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP 92), San Francisco, 1992, pp. I-9–12.
[2] V. Steinbiss, H. Ney, X. Aubert, S. Besling, C. Dugast, U. Essen, D. Geller, R. Haeb-Umbach, R. Kneser, H.-G. Meier, M. Oerder and B.-H. Tran, Philips J. Res., **49**(4) 317–352 (1995).
[3] M. Oerder and H. Ney, Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP 93), Minneapolis, 1993, pp. II-119–122.
[4] K.S. Fu, Syntactic Pattern Recognition and Applications, Prentice-Hall, Englewood Cliffs, 1982.
[5] A. Demmel, Interpretation deutscher Zeitausdrücke, Master's thesis, 1991, University of Kaiserslautern, Germany.
[6] H. Aust and M. Oerder, Proceedings of the ESCA Workshop on Spoken Dialogue Systems, Aalborg, 1995, pp. 121–124.
[7] B. Bly *et al.*, Proceedings DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, San Mateo, 1990, pp. 136–140.
[8] R. Moore and A. Morris, Proceedings DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, San Mateo, 1992, pp. 61–63.
[9] J.F. Allen, Proceedings DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, San Mateo, 1992, pp. 5–6.
[10] E. Gerbino *et al.*, Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP 93), Minneapolis, 1993, pp. II-135–138.
[11] C.T. Hemphill, J.J. Godfrey and G.R. Doddington, Proceedings DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, San Mateo, 1990, pp. 96–101.