Spoken language systems — beyond prompt and response

P J Wyard, A D Simons, S Appleby, E Kaneen, S H Williams and K R Preston

Spoken language systems allow users to interact with computers by speaking to them. This paper focuses on the most advanced systems, which seek to allow as natural a style of interaction as possible. Specifically this means the use of continuous speech recognition — natural language understanding to interpret the utterance, and an intelligent dialogue manager which allows a flexible style of 'conversation' between computer and user. This paper discusses the architecture of spoken language systems and the components of which they are made, and describes both a variety of possible approaches and the particular design decisions made in some systems developed at BT Laboratories. Three spoken language systems in the course of development are described — a multimodal interface to the BT Business Catalogue, an e-mail secretary which can be consulted over the telephone network, and a multimodal system to allow selection of films in the interactive TV environment.

1. Introduction

No science fiction image of the future is complete without the ever-present personable computer that can understand every word which is said to them. In spite of these popular media images, the goal of natural interaction between humans and machines is still some way off.

Interactive voice response (IVR) systems, which provide services over the telephone network, have been available since the mid-1980s. Initially they were restricted to interactive TouchTone(R) input with voice providing the response to the user. The use of such services was therefore limited to the population with TouchTone keypads. More recently applications using automatic speech recognition (ASR) have been developed. These often simply allow the option of spoken digit recognition as an alternative to keypad entry, thus allowing the service to be launched even in areas where TouchTone penetration is poor. Moving on from such systems the words which are spoken can be matched to the service. This allows these ASR-based services to be more user-friendly than their TouchTone counterparts because the user can directly answer the question: 'Which service do you require?' with 'weather' or 'sport' rather than 'for weather press 1 for sport press 2', etc. However, they still rely on selection from a predetermined menu of items at any point in the dialogue.

More sophisticated services are now becoming possible using emerging larger vocabulary speech recognition technology. However, it is not sensible to simply extend the menu-based approach to accommodate larger vocabularies. Although well-engineered simple applications may be easy to use, more advanced services are likely to have complicated menu structures. If information can only be provided one item at a time, using a 'prompt and response' dialogue, rigid interaction styles may steer the user through a complex dialogue. This can result in the user becoming lost, or ending up with the wrong information. These problems are particularly significant for inexperienced users. On the other hand, experienced users may become bored by the large number of responses needed when they know exactly what they want. The menu-based structure required by systems which rely on isolated word input is often the limiting factor for new services. This limitation of the user interface is one of the greatest barriers to the usability of many IVR services.

Moving beyond the menu-style interaction towards conversational spoken language will allow users to express their requirements more directly and avoid tedious navigation through menus. This approach will also allow the user to take control of the interaction rather than using the more common 'prompt and response' dialogue.

BT is interested in the development of spoken language systems (SLS) to provide a key competitive advantage. SLSs allow users to interact with computers using conversational language rather than simply responding to system prompts with short or one word utterances. With the rapid increase in competition, service differentiation becomes a key factor in gaining market share. Systems

which allow users 24-hour remote access to information provide a very useful service for people who are in different time zones, or away from their office, or who need information immediately during unsocial hours. SLSs can be used to automate such services and also those which currently require human operators, thus freeing their time to deal with difficult situations where more complex, or more personalised advice is needed.

Current trends in information networking and the phenomenal growth of the Internet bring their attendant problems for our customers in keeping up with technology, finding what they need, and using information to their best advantage. Spoken language system technology can greatly enhance our customers' ease of access to information, thus increasing network revenue through new and increased usage. Systems which combine several modes of input and output, such as speech, graphics, text, video, mouse-control, touch and virtual reality, are known as multimodal spoken language systems. These allow far greater freedom of expression for users who, as a result, should feel more comfortable and less as though they are 'talking to a computer'. They are able to point, use gestures, speak, type; whatever comes most naturally to them. Spoken language systems will become increasingly important in the near future as progress in technology becomes more widely available.

The goal is to be able to build systems which are not restricted only to those motivated users who are prepared to spend time learning the language the machine understands. These new systems can be used by anyone who wants occasional access to a particular service. They will also help the user successfully gain the information or service they require by simply calling a number and asking for what they want. In fact, the aim is to put back some of the intelligence which existed in the network 50 years ago when a user simply lifted the handset and asked to be connected to the service or number required.

This paper discusses the design and implementation of spoken language systems and is organised as follows. Section 2 gives an outline of the architecture of an SLS. Section 4 describes the components of an SLS in some detail, giving concrete examples from current systems. Section 3 discusses some of the systems currently under development at BTL. These include a multi-modal system for access to the BT Business Catalogue, a speech-in/ speech-out system for remote e-mail access and a system for accessing information about films. Section 5 discusses future work which needs to be carried out to improve the quality and usability of SLSs, and section 6 draws some conclusions.

2. System overview

This section outlines a typical spoken language system architecture, from the information processing point of view (platform and inter-process communication issues are not dealt with to any great extent in this paper.) The architecture and the key processing components are outlined.

The most basic form of SLS, a speech-in/speech-out (rather than multimodal) system, requires at least the following major components (described briefly below and in more detail in section 4).

- Speech recognition — to convert an input speech utterance to a string of words.
- Meaning extraction — to extract as much of the meaning as is necessary for the application from the recogniser output and encode it into a suitable meaning representation.
- Database query — to retrieve the information specified by the output of the meaning extraction component. Some applications (e.g. home banking) may require a specific transaction to occur. Many applications may be a mixture of database query and transaction processing.
- Dialogue manager — this controls the interaction or 'dialogue' between the system and the user, and co-ordinates the operation of all the other system components. It uses a dialogue model (generic information about how conversations progress) to aid the final interpretation of an utterance. This may not have been achieved by the 'meaning extraction' component, because the interpretation relies on an understanding of the conversation as a whole.
- Response generation — to generate the text to be output in spoken form. Information retrieved by the database query component will be passed to the response generation component, together with instructions from the dialogue manager about how to generate the text (e.g. terse/verbose, polite/curt, etc).
- Speech output module (text-to-speech synthesis or recorded speech).

At its simplest, processing consists of a linear sequence of calls to each component, as shown in Fig 1. A typical output of each stage from an application which accesses the BT Business Catalogue is shown. It is not necessary to understand the output of the 'meaning extraction' component in detail to realise that meaning extraction can be a non-trivial exercise. The simple linear sequence shown in Fig 1 is, in general, too inflexible. It is better if the dialogue manager is given greater control, to call the other components in a flexible order, according to the results at

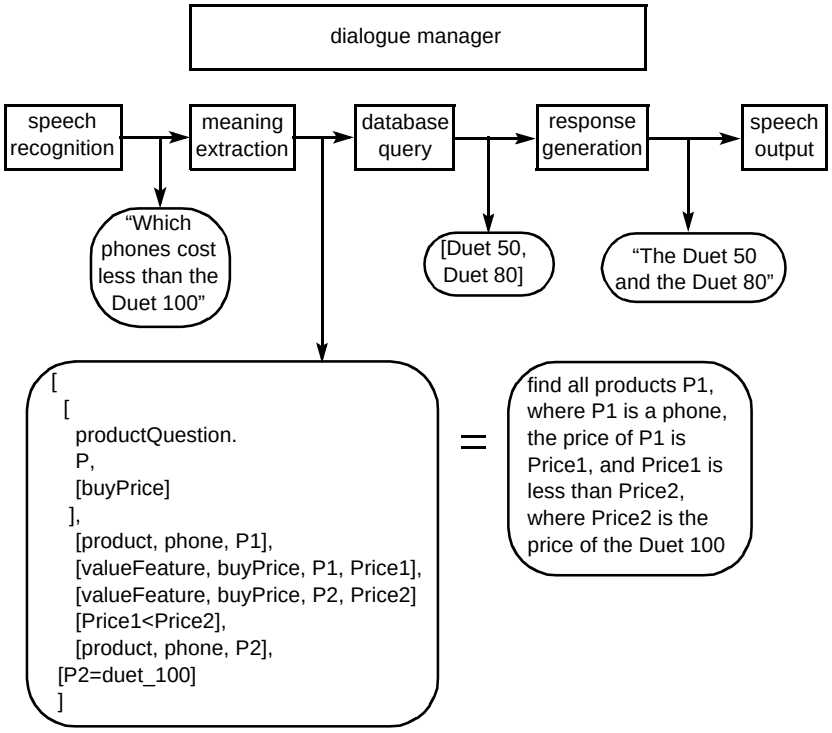


Fig 1 Example of a linear process flow in a spoken language system.

each stage. This leads to an architecture of the type shown in Fig 2.

The need for this more flexible architecture is illustrated by the processing sequence in Fig 3 which shows the dialogue manager as control centre, calling each component in an order determined by the results of processing at each stage. Although every processing stage is passed through the dialogue manager, this is not included in the sequence unless some non-trivial decision or action is taken. The example given in Fig 3 is largely driven by limitations of the recogniser, but the need for this sort of flexible architecture goes far beyond this. It will eventually enable the dialogue manager to act in an intelligent manner, co-ordinating the components and combining their outputs in a nonlinear manner.

So far in this section, the discussion has covered speech in/speech out systems. However, systems such as the BT Business Catalogue access system (see section 3.1) are multimodal and require a screen and a means of inputting text and mouse clicks and outputting text and graphics. These components must be added to the architecture shown in Fig 2 and the dialogue manager and response generator must be upgraded to deal with the extra modalities. However, most of the discussion of this section applies equally to multimodal systems.

3. Example systems

In this section three spoken language systems under development at BT Laboratories are described:

- access to the BT Business Catalogue, known as BusCat — this was the first multimodal continuous speech input spoken language system,

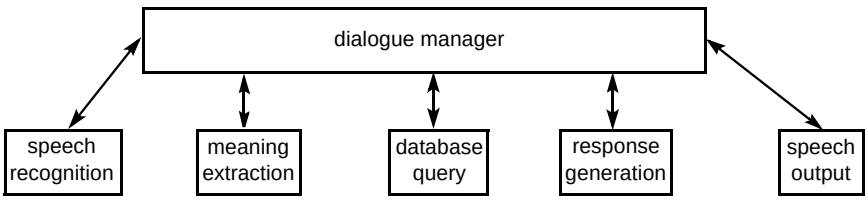


Fig 2 Role of a dialogue manager in a spoken language system.

- an e-mail access system, which is speech in/speech out only, but has the conversational features described in this paper — it is also a dial-up service over the telephone network,

- a film access system, in which users will be able to select films and videos using continuous speech and button pushes on a remote control handset — this system is targeted at the interactive TV environment.

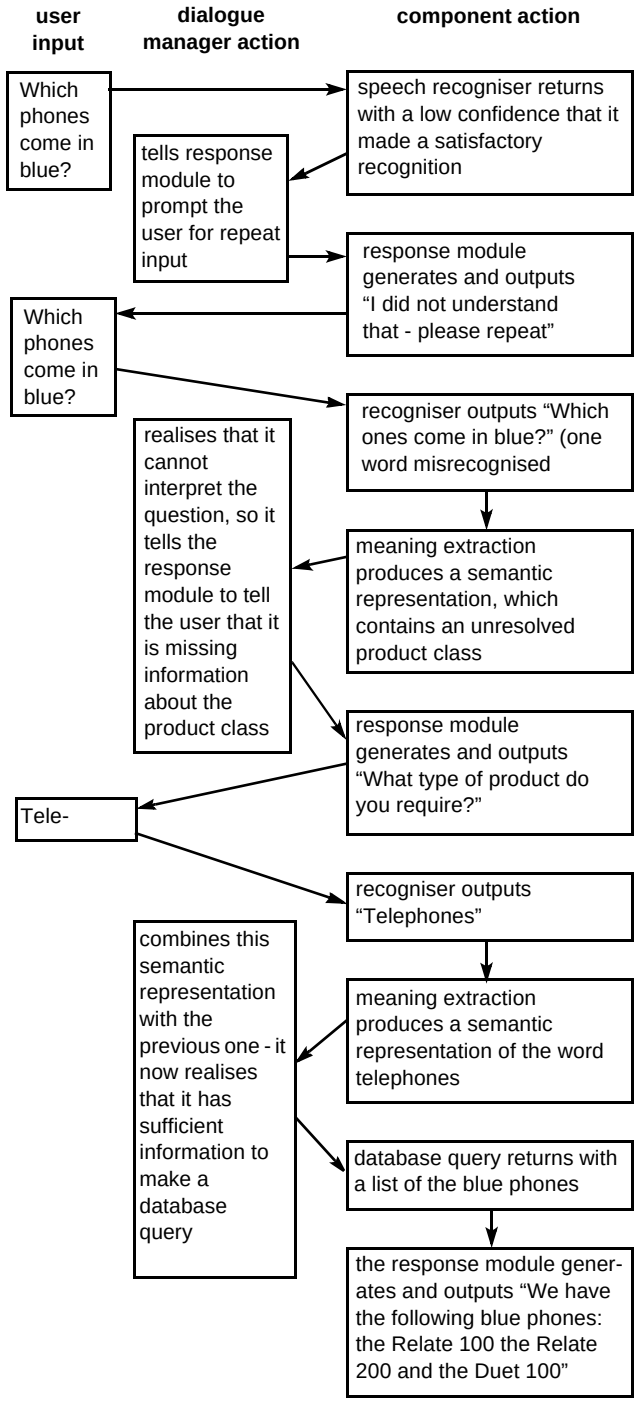


Fig 3 Nonlinear process flow in spoken language systems.

3.1 BusCat

The SLS BusCat provides direct access to a subset of the BT Business Catalogue, which covers a range of products such as telephones, answering machines and phone systems. The user has a screen displaying a Netscape WWW browser and speech input/output facilities. All the normal WWW browser features are present, such as the ability to click on links to other pages, and a display consisting of mixed text and graphics (see Fig 4).

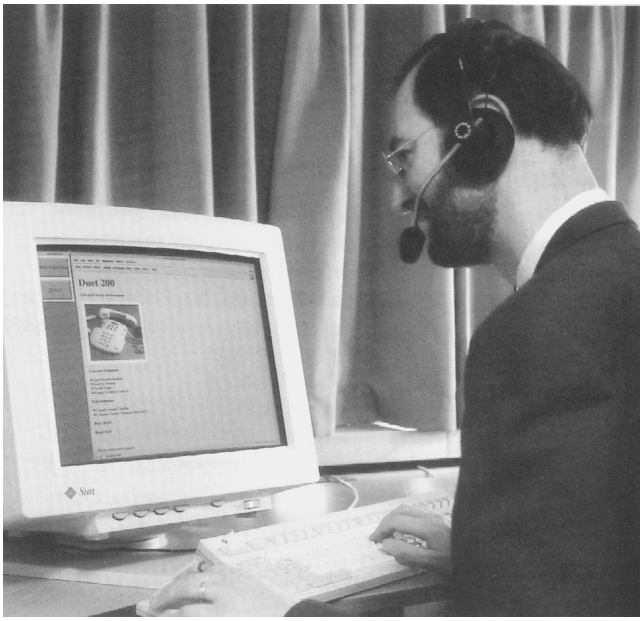


Fig 4 The SLS BusCat system in use.

Additionally, in this system users may use continuous speech input, type questions into a free-text window, and listen to speech output generated by a text-to-speech (TTS) system. This multimodal interface enables users to request specific information about the products in the catalogue, or to browse through the catalogue.

The overall structure of the system is shown in Fig 5. The system can cope with multiple simultaneous users.

In addition to its internal knowledge bases, the system has the capability to access external databases across a network. One application for this might be to provide a multimodal interface for such databases. Another is to allow the internal knowledge bases to be periodically updated from an external database.

The speech recogniser used is BT's Stap recogniser [1], and the text-to-speech system is BT's Laureate [2] system.

The example in Table 1 gives a flavour of what it feels like to interact with the system. Here the user is already logged on to the system. From each WWW page there is a choice of:

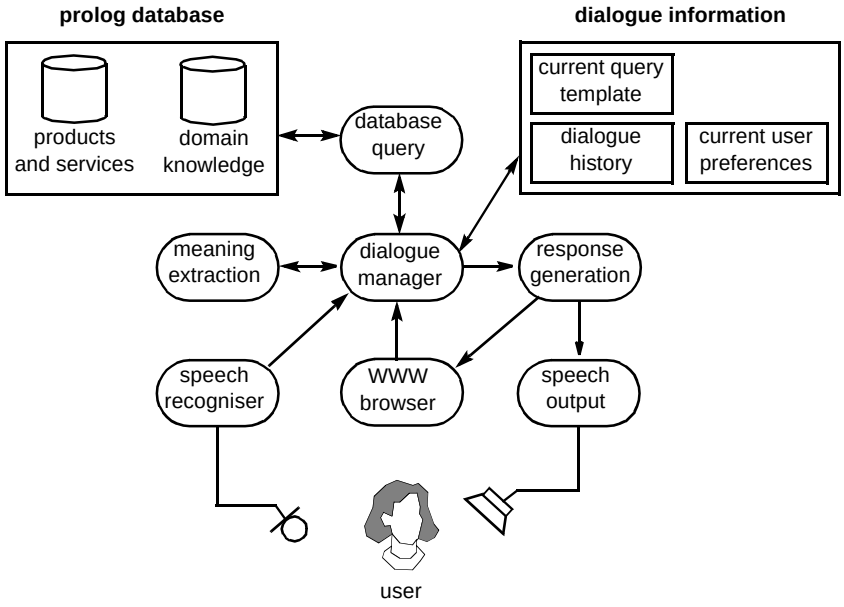


Fig 5 Architecture of BusCat.

Table 1 An example session with BusCat.

User input	System response
'What is on-hook dialling?'	Textual (and optionally spoken) explanation of on-hook diallin: 'Time spent waiting for someone to answer the phone can often be lost time. But with this feature, you can dial without picking up the phone handset, leaving you free to carry on with something else until the second your call connects,' and a list of five phones which have this feature: Vanguard 10e, Relate 200, Relate 300, Relate 400, Converse 300.
'Which phones have on-hook dialing and cost less than 60 pounds?'	Text: 'The following products meet your requirements,' and a list of four phones, each with a small picture, a short description and a price (Vanguard 10e, Relate 200, Relate 300, Converse 300).
'Which ones come in grey?'	Text: 'The following products meet your requirements,' and a list of three phones, each with a small picture, a short description and a price (Ganguard 10e, Relate 200, Relate 300).
The user clicks on the link next to the picture of the Relate 200.	The system responds with a large picture of the Relate 200, a ful description including all its features and a price.

- speaking to the system,
- clicking on a link,
- typing into the free-text field.

In the interaction the user wants to know what on-hook dialling is. Having received an explanation of this feature, he decides he wants a phone with on-hook dialling which costs less than #60. Then he remembers he also wants it in grey to match his living room. He finally selects the Relate 200 telephone.

3.2 E-mail access

BT is very interested in the mobile telephony market. Speech-only natural language systems are very attractive to this market because people want to be able to keep in touch while on the move. They are likely to only have a mobile phone with which to do so. Speech access to information can have an added safety advantage over TouchTone interfaces as well as helping to improve the usability of the service.

E-mail has been chosen as the vehicle for producing a speech in/speech out natural language system because the information source (the users own e-mail) is readily available. An existing system allows selected users to access their e-mail using TouchTone commands, thus providing information on whether natural language access confers real benefits to the users. It will certainly enable queries which are not possible currently, such as: 'Do I have any messages from John?'

The main differences between the e-mail service and the BusCat service are as follows. The e-mail service uses telephony speech only, rather than a multimodal system. The dialogue is more robust to errors and will allow the user or the system to take the inititive whereas the Business Catalogue system is based on the user taking the initiative. The e-mail system architecture is being designed to

decouple domain-independent and domain-dependent parts of the system, and the same approach is being used for the film access system (see section 3.3). This means that in the future there will be a good level of confidence in being able to produce spoken language systems for new domains more quickly than at present.

An example of a dialogue the system might have with the user is shown in Table 2.

Table 2 An example session with an e-mail service.

User	System	Note
(dialls up system)	Good morning Alison. You have ten new messages and 20 old messages. Your first new message is from Mike Harper, regarding, Thanks for demo.	The default startup is to tell the user how many old and new messages there are and to start listing the new ones.
Read it.	Message is: 'Thank you for your demo yesterday. The customers were most impressed.	
Forward it to Peter Smith and Sandra	Sandra Jones is already on the distribution list	Resolution of partially specified name. Check on preconditions for forwarding
	Forwarding message to Peter Smith	
	Next message is from Colin North, regarding, Team Meeting.	System assumes you want to carry on listing new messages unless you tell it otherwise.
Forward to Language Group	Forwarding message to all the people on your distribution list called 'Language Group'.	Does not confirm on forwarding a message.
	Next message is from Laurence Cooper: Plans for Workshop	
Let's hear it.	Message is: 'Hi Alison, Have you completed the plans yet?'	
Reply	I have four possible responses, yes, OK, no, or I'll call you.	Choice of several stock replies
Tell him yes	Reply to Laurence Cooper: Text is 'Yes'. Please confirm.	Confirms before sending a reply
OK	Reply sent. Next message is from...	
Stop.	... OK. What would you like?	Detection of user speaking terminates prompt. Listing new messages ended by user.
Have I any recent messages from Steve Brown or Steve Clarke	In the last week you have received 5 messages from Steve Brown and 4 from Steve Clark	Slightly more complex user query. System makes explicit what it thinkgs of as 'recent'.
List the ones from Clark.	Message 1 is about Meeting on the 31st Message 2 is about Agenda for team meeting	Resolution of partial specified name.

3.3 Film access for interactive multimedia services

Interactive multimedia services (IMS) are currently being implemented by BT in Colchester and Ipswich [3]. The usability of the interface is still constrained to navigation using a TV remote control. This means that multiple layers of menus need to be traversed to get to the information required. Spoken language access would allow users to go straight to the information they are searching for, without requiring them to learn complex navigation procedures.

The video-on-demand subset of the IMS, which consists of over 4000 hours of material, including films, educational programmes, children's programmes, etc, was chosen. The SLS will allow users to give instructions such as: 'I want a comedy film starring Harrison Ford'. Part of the benefit of developing such a system, is to ensure that the generic SLS framework is truly domain independent.

There is currently a text-based interface to the Internet movie database [4]. This allows users to enter queries such as: 'Tell me the ratings of comedy movies starring Harrison Ford'. The system performs the meaning extraction using a caseframe parser (section 4.2). This allows it to pick out the salient information from among extraneous words.

It seems likely, from human analysis of typical queries about films, that this method is suitable.

An issue yet to be addressed is how to best reconcile the advantages of using speech, with the limitations of current recognition technology. This is clearly illustrated in the present example, since the text-based interface can query the database of over 50 000 films and 100 000 cast names. No speech recogniser yet built can cope with this range of vocabulary. The obvious solution is to restrict the size of the database. A possible step in the right direction would be to couple the 'meaning extraction' component and recogniser much more closely, so that meaning extraction and recognition happen simultaneously. This might enable the recogniser to cut down the vocabulary size 'on the fly'. For example, given the input sentence: 'Which comedy movies star Burt Lancaster,' it could be established straightaway that the user was talking about comedies, then only about cinema films, and finally that the user was only interested in an actor. Therefore, by the time the recogniser gets to the name 'Burt Lancaster,' the number of possible words has reduced considerably.

This is the subject of further research and is discussed in more detail in the next section.

4. Components of a spoken language system

4.1 Speech recognition

The job of a speech recogniser ids typically thought of as converting an utterance of speech into a string of text. The internal workings of speech recognisers are explained in some depth elsewhere [5]. This section looks at the recogniser's place within an SLS, and, in particular, at the language model (LM) the recogniser uses and the form of output that it provides.

A language model embodies information about which words or phrases are more likely than others at a given point in a dialogue.

One might imagine that in a system that accepts fluent language, for example an automated travel agent, the speech recogniser might need only one language model, that of the entire English language. It could then recognise anything that anyone said to it (assuming they are speaking English) and could inform the dialogue manager accordingly. Speech recognition is not yet accurate enough and a model of the entire English language does not exist. Instead, to get a working system, the recogniser must be given as much help as possible. It must be given hints about what the user is likely to say next to improve the chances of correctly recognising what has been said. If the dialogue manager knows that the customer wants to go on a cruise and has just asked them where they would like to go, it should prime the recogniser to be expecting a response that may well concern one of a number of specified cruise ports and, by the same token, is unlikely to have anything to do with backpacking in Nepal.

Recognisers use a language model to hold this information. There are a number of ways that the dialogue manager can update the information that the recogniser is using. The simplest option is just to tell the recogniser which of a predefined set of language models to use. Then there is a range of possibilities for updating or modifying predefined language models. Certain portions of a grammar can be made more likely than others, e.g. phrases to do with the time of day might be expected at one stage in the dialogue, while requests concerning holiday destinations might be more likely at another, and the language model can be adjusted accordingly. Alternatively, the recogniser might have a grammar, a portion of which allows the sequence 'from <airport> to <city>' where the range of possible airports and cities is specified by the dialogue manager only immediately prior to recognition. This could be dependent, say, on which country is under discussion.

The following subsections discuss in more detail:

- language models,
- perplexity of a language model,
- advantages and disadvantages of language models,
- loading language models into the recogniser,
- output from the recogniser.

4.1.1 Language models for the recogniser

The primary knowledge source for the speech recognition component is a set of statistical models, known as hidden Markov models or HMMs, which encode how likely a given acoustic utterance is, given a string of spoken words. A recogniser can decode a speech utterance purely on the basis of this acoustic-phonetic knowledge, and this is basically what happens in the case of single isolated-word recognition. However, in the case of recognising a string of words (which form part of a spoken language), the recogniser can use a second knowledge source, namely the intrinsic probability of the given string. This second knowledge source is known as the language model.

To take a classic example, a given utterance may have almost equal acoustic-phonetic probabilities of being 'recognise speech' or 'wreck a nice beach'. However, the intrinsic probability of the first string is likely to be higher than that of the second, particularly if this utterance came from the domain of a technical journal on speech technology.

This can be expressed mathematically as follows. Let $\boldsymbol{X}$ be the acoustic utterance and let $\boldsymbol{S}$ be the sentence to be recognised. The task is to find the sentence $\boldsymbol{S}$ for which the posterior probability $p(\boldsymbol{S}|\boldsymbol{SX})$ is a maximum. Using Bayes' rule this can be rewritten as a requirement to find:

$$\underset{\boldsymbol{S}}{\text{argmax}} \{p(\boldsymbol{S}) * p(\boldsymbol{X}|\boldsymbol{S})\}$$

In this formula, $p(\boldsymbol{S})$ is the prior probability of the sentence according to the language model, and $p(\boldsymbol{X}|\boldsymbol{S})$ is the conditional probability of observing the acoustic utterance given the sentence (encoded in the HMMs of the recogniser).

In general, the language model in a recogniser consists of all the language information to which it has access in order to help constrain the recognition, by making certain word strings less likely than others, or indeed impossible. This language information may be the same information as used in the 'meaning extraction' component of the system (see section 4.2). For example, when the grammar in the meaning extraction component is compiled down to a finite state network (FSN) for use in the recogniser. Figure 6 gives an example of a recogniser FSN. More commonly, the language information in the recogniser is represented by a statistical model, possibly derived from the same corpus

data as the language information in the 'meaning extraction' component, but generally only employing recent context (one or two words). This type of model is known as an n-gram model. Such models are easy to integrate with the standard acoustic decoding architecture in speech recognisers, but ignore some of the available language information which might improve accuracy.

N-gram models can be word n-grams, class n-grams or a hybrid of the two. n is typically 2 or 3, and these models are referred to as bigrams and trigrams respectively. A word bigram model gives the probability of all possible next words on the basis of the current word only, i.e. it is of the form $p(w_2|w_1)$. A trigram model is based on two words of context, $p(w_3|w_1w_2)$. The probability of a sentence such as 'delete that one' is obtained by multiplying the probabilities for each word, given its predecessor(s), e.g. for a bigram model:

$p(\text{Sent}) = p(\text{delete}|\text{start}) * p(\text{that}|\text{delete}) * p(\text{one}|\text{that}) * p(\text{end}|\text{one})$

A class n-gram model is one in which the words are grouped into classes before computing the n-gram statistics. These classes may be syntactic categories such as nouns or verbs, hand-crafted semantic categories (e.g. all days of the week might be grouped together, all names of people, etc), or automatically learnt classes. The probability of a word w_3, given two previous classes, c_1, c_2, is given by:

$$p(w_3|c_1c_2)=p(c_3|c_1c_2) * p(w_3|c_3)$$

The advantage of a class n-gram model is that there are fewer parameters to estimate. The disadvantage is that a word is predicted only on the basis of the class history, so some distinctions which may be present in the training data are lost; for example, 'Monday morning' may be more likely than 'Monday afternoon', whereas for all the other days of the week, 'DAY morning' and 'DAY afternoon' may be equally likely.

An FSN gives all possible word sequences in the language model, and it may have probabilities attached to the transitions if reliable statistics for these can be obtained. For example, a fragment of a FSN LM is shown in Fig 6.

Both n-gram and FSNs give a probability for the next word wn given a word history $w_i.....w_{n-1}$. In the case of n-grams, the history is only one or two words, as stated above (e.g. $i = n - 2$ for a trigram model). In the case of FSNs, the history extends back to the start of the sentence (i.e. $i = 1$)

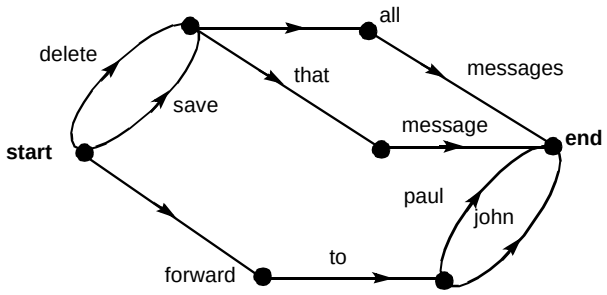


Fig 6 A recogniser finite state network.

4.1.2 Perplexity

In some cases, the constraints imposed by the language model are so great that the job of acoustic decoding becomes much easier than one would have imagined from the size of the vocabulary. This is because the LM is effectively ruling out many possible strings from the search space in which the acoustic decoder is operating. It is possible to calculate a figure called the perplexity associated with a given LM, which is a measure of how difficult the acoustic matching task for the recogniser is. Usually, the lower the perplexity, the higher the recognition accuracy. Perplexity can be roughly defined as the average branching factor, i.e. the average number of words which are allowed to follow a given word. Perplexity may either be calculated intrinsic to the LM (for an FSN, perplexity is literally the average branching factor of the FSN), or with reference to a test corpus, which is the way one usually calculates the perplexity of an n-gram LM. In the latter case, perplexity is defined as the reciprocal of the geometric mean of the probability of the next word, for all words in the test corpus. Intuitively, perplexity is low if the language model tends to know what is coming next, which means that the next word probabilities are relatively high. The formula for test set perplexity is:

$$PP = \left[\prod_i P(w_i)\right]^{-1/N}$$

where $P(w_i)$ is the probability of the ith word,
N is the number of words in the test set,
and the index i runs over each word in the test set.

Typical values of perplexity may range from about 40 to several hundred in current research systems.

4.1.3 Advantages and disadvantages of FSNs and n-gram LMs

An advantage of FSNs is that they have a fairly low perplexity, i.e. the number of legal following words is usually a small subset of the entire vocabulary. They can

also be constructed manually, before a proper training corpus for the domain exists.

One disadvantage of FSN language models is that they can easily be over-restrictive, ruling out many perfectly valid input strings, simply because of the impossibility of anticipating all the different things users will want to say. This disadvantage can be mitigated by making the FSN more flexible, with liberal use of optional phrases (as in BusCat), or going some way to turning it into an unconstrained phrase network, where any phrase (a few words) can follow any phrase. This of course has the effect of increasing the perplexity, and at a certain point it becomes preferable to move to an n-gram LM. A second disadvantage of FSN language models is that as one moves towards a broader, more habitable user language, they very quickly become large, making the recognition time unacceptably long. The FSNs themselves become cumbersome and difficult to maintain.

N-gram language models, on the other hand, usually have a higher perplexity than FSNs, because all words may be legal followers of the current word, albeit many of them having low probabilities.

The disadvantage of n-gram LMs is that they are not as constraining as FSN LMs, which means that if the acoustic match accuracy is not good enough, the output of the recogniser may be so poor that the meaning extraction component has little chance of interpreting it, however robust its algorithm.

4.1.4 Loading the LM into the recogniser

The LM(s) may be loaded into the recogniser:

- once at the start of an application,
- repeatedly, according to where the user is in the dialogue immediately prior to recognition,
- during a recognition by the recogniser.

These three possibilities represent increasing degrees of sophistication. In the first case, there is just one LM for the whole application. In the second case, use is made of the fact that at different points in the dialogue the relative probabilities of different words will change greatly. For example, if the system asks: ‘Which day do you wish to travel on?’, there is higher probability than normal that the user’s answer will contain a day of the week. Similarly, if it has already been established that the user wishes to travel from Ipswich, there is a higher probability than normal that the destination will be somewhere in the Anglia region.

In the third case, the LM is changing ‘on the fly’, during the decoding of the speech utterance. For example, if the first words in the sentence are ‘I want to go from Ipswich to..’, then the LM can increase the probabilities of stations which have direct trains from Ipswich before recognising the rest of the sentence.

The LM(s) passed to the recogniser may be complete models or modifications to the current model. A modification to a current model might consist of:

- a list of words to be added or deleted,
- modified probabilities for words which are already within the language model.

4.1.5 Output from the recogniser

If the job of a speech recogniser is to convert a speech utterance into a string of text, then, when someone says: ‘What time does the flight leave?’ it is hoped that the recogniser might come up with the string: ‘What time does the flight leave’. In practice the recogniser’s best guess may not be correct, but it may be able to give a number of scored alternative possibilities which the analysis module might use if the top choice does not make sense, for example:

1 ‘what time does the white leaf’ 1245.6

2 ‘what time does the flight leave’ 1250.1

3 ‘what time does a flight leave’ 1252.3

4 ‘what time did the flight leave’ 1270.1

5 ‘what time did a flight leave’ 1272.3

The analysis module may then use its own rules to re-rank the list of possibilities or extract those portions that carry useful information (perhaps taking account of the recogniser scores or perhaps ignoring them).

There are, however, significant drawbacks in representing the recogniser output as a ranked list of word strings. The list can become very long indeed, containing large numbers of sentences that only differ by one or two words.

A much more compact representation of the same information can be achieved through the use of directed graphs (see Fig 7).

A third option that is sometimes used is the word lattice (see Fig 8). This consists of a lattice with entries, each of which specifies a possible word, its start time and its end time (but not which other words it follows or precedes). The benefit of the word lattice is also its weakness. It can offer in a compact form for storing a very large number of candidate sentences — sentences can be generated by connecting any sequence of words such that one word starts where the previous one finishes. Unfortunately this allows one to generate sentences that the recogniser did not consider very likely, e.g. ‘what time did a flight leaf’.

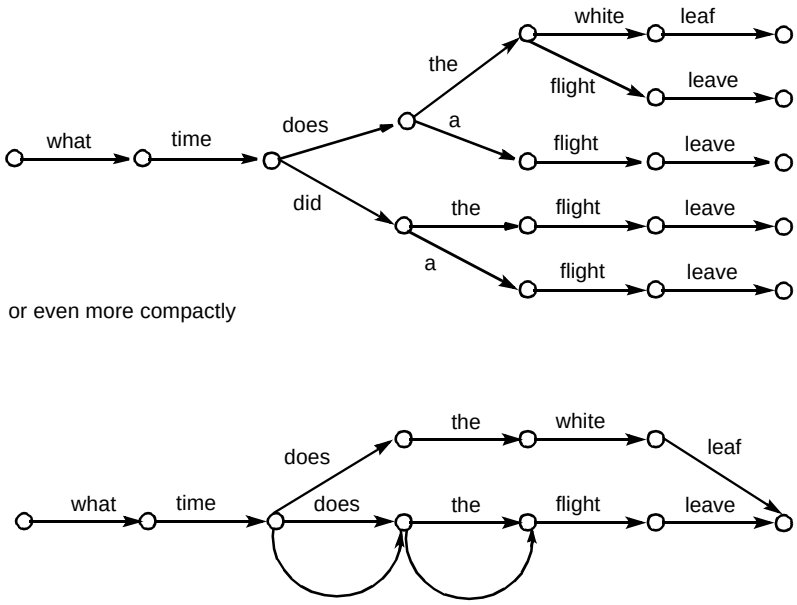


Fig 7 Directed graph for recogniser output.

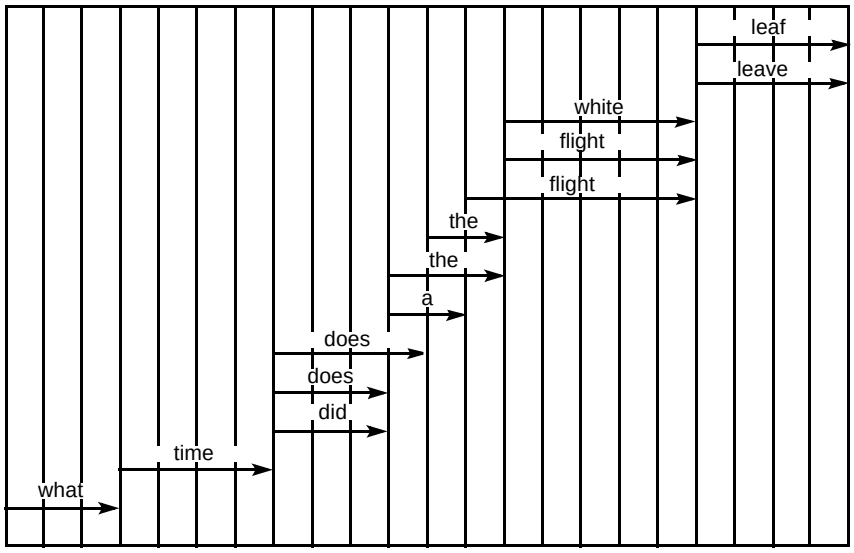


Fig 8 World lattice for recogniser output.

4.2 Meaning extraction

4.2.1 Purpose of the meaning extraction component

The input to this component is a representation of the user's utterance that is the output from the speech recogniser. This representation may take one of several forms as described in the previous section. The meaning extraction component of the SLS converts the input to a representation of the meaning of the utterance, which is used to determine the next step in the dialogue.

The user's utterance will contain several pieces of information, some of which will need to be represented in the output from the 'meaning analysis' component. One piece is the type of utterance that the user made, e.g. an instruction, a statement or a particular kind of question. Another concerns the expectations present in the user's utterance. These are often expectations about the result of a query and are not usually intended to be used as constraints on the query itself (such as the use of a plural form to indicate that the user is expecting more than one item to satisfy a request). A third piece of information consists of the entities referred to in the user's utterance, and the

relationship between them. These will form the basis of the constraints on the database query. The purpose of the 'meaning extraction' component is to make some of this information explicit so that it may be more easily processed by subsequent components.

Although the role of this component can be described simply enough, the task is daunting. There are at least two main difficulties that need to be overcome during meaning extraction in SLS:

- ambiguity,
- ill-formed input.

Ambiguity [6] may be present in words as lexical ambiguity (e.g. 'saw' can either be a noun or a verb) or sense ambiguity (e.g. 'bank' may be a place in which to store money, or from which to fish), or it can be present in the relationship between phrases as structural ambiguity. To take the ubiquitous example:

'The man saw the boy with the telescope in the park.'

This sentence is structurally ambiguous in what was seen, the instrument of seeing, and the location of the person seeing. Often, this type of ambiguity cannot be resolved by the meaning extraction component, and has to be maintained in the meaning representation, to be deciphered by later components.

Ambiguity is inherent in all natural language, while ill-formedness can be caused by misrecognition in the speech recogniser, or by the user making an ungrammatical utterance (as defined by the system). Speech recognisers are becoming increasingly competent, but they still mishear quite often. This is particularly a problem with recognisers which use an n-gram language model which are less restricted in the ordering of the words they can recognise. For example, in a 'hotel enquiry' domain [7], the parser was faced with input such as:

'do the noise outsiders use your lake'

when the user actually said:

'there is a noise outside that keeps me awake'

It is doubtful that any meaning extraction component would be able to cope with such a discrepancy between the actual utterance and the recognised one, but there are mechanisms by which simpler ill-formed utterances can be analysed [8]. These will be explored in more detail below, but essentially they attempt to pick out those bits which can be analysed, and put them together in the best way possible.

If meaning extraction is viewed as performing a translation between an input text and its meaning, then, quite apart from the two particular problems described above, the difficulty of defining the translation is faced. This is a many-to-many mapping, since many English sentences can have essentially the same meaning (in the sense that they are true in the same situations) and some sentences can have more than one meaning. For instance, the following two sentences, to all intents and purposes, mean the same, but have a different structure:

'Romeo loves Juliet' and 'Juliet is loved by Romeo'

Assuming that all of the above problems can be overcome, a suitable representation for the output meaning must be chosen. The issues to be considered are:

- suitability for successive stages of processing — whether the representation encodes all the information necessary for further processing,
- explicitness versus conciseness — whether one representation can encode all unresolved ambiguity, while maintaining a clear meaning,
- extendibility — whether the representation will only cope with the particular sentences chosen for development.

The representation generally depends on the technology used, but various logics [9] are popular forms of representation. This is because they are well-understood, and fully-specified. Perhaps more importantly, a meaning represented in logic can be reasoned with, using the proof calculus of that logic. This is useful in database query systems [10]. The other main representation is to use a frame which encodes predefined meanings (such as the act of 'seeing') [11]. These frames will take arguments to fill in the details ('who', 'what', 'where', etc).

The rest of this section is devoted to a particular representation based on a logical representation. This representation, which is referred to as an extended logical form (ELF), has three fields corresponding to three types of information present in the user's utterance. An example ELF, representing the utterance 'read the message from John' is given in Table 3.

Table 3 Fields in an extended logical form representation.

Field name	Field content
Type	read(A),
Expectations	salient(A), singular(A)
Entities and relationships	message(A), named(B john), from (A,B)

The type field is used to indicate the type of the utterance, such as imperative, indicative, question, among others.

The expectations field contains a list of constraints that are the user's assumptions. In the example above, the user is presupposing that there is only one message from someone called John, therefore 'singular(A)' will be an item on the list of expectations. The constraint, 'salient(A)', is used to indicate that the user has assumed that there is some specific (usually recently mentioned) message that is the object of the instruction. This contrasts with 'read a message from John' where there would not be a 'salient(A)' constraint on the list.

The entities and relationships field of the ELF contains a list of conjunctive constraints which express the entities referred to in the user's utterance and the relationships between them.

The frame representation and the logical representation are indicative of the separation between different meaning extraction technologies. The logical form of representation tends to be used by grammar-based parsers, while the frame form tends to be used by caseframe parsers. These different types of parser will be described in the next two sections.

4.2.2 Grammar-based parsers

Traditionally, schoolchildren who are given the exercise of analysing English sentences according to a prescribed grammar of English have been doing parsing [12]. Parsing has the purpose of assigning a structure to an input utterance. This can be done by hand, but is now more often done using an automatic parser. More precisely, what has just been described would be a syntactic parser, performing a syntactic analysis. Here, the sentence is being analysed according to a syntactic grammar [13], where the relationship between nouns and verbs, for example, is made explicit. Syntactic analysis gives no representation of the meaning of the sentence. Indeed, the following two sentences would be assigned the same syntactic structure [14]:

> 'Bright red buses drive quickly' and 'Colourless green ideas sleep furiously'.

Of course, they clearly have a different meaning, and semantic analysis [15] is concerned with providing such a meaning, solely from the combination of the meaning of the individual words.

Grammar-based parsers take a linguistic grammar of, say, English, and assign a structure to the sentence based upon that grammar. Two stages are necessary -- first, assign each word a part of speech (POS), such as noun, and, secondly, from these POS apply the grammar rules.

So, given the phrase:

'the main'

the lexicon which contains the following rewrite rules:

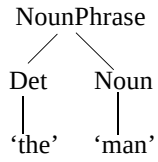
Det → 'the'

Noun → 'man'

and the grammar rule:

NounPhrase → Det Noun

the parser would build the syntactic structure:



Syntactic analysis is not sufficient for meaning extraction as mentioned previously, and a semantic analysis stage is needed. This can either take place after the syntactic analysis or concurrently with the syntactic analysis. Indeed, semantics is often built up in parallel with the syntax, using the rule-rule hypothesis [15]. This basically states that syntactic and semantic rules should only exist in pairs, the semantic analysis being built by 'piggy-backing' on the syntactic analysis. This means that every part of the sentence contributes to its overall meaning, and that fine-grained syntax produces fine-grained semantics. Extremely complex grammars can be written [16], encapsulating fine points of language. These details may be important in some systems, and are thus represented in the final meaning representation.

The primary problem with grammar-based parsers, from the point of view of SLSs, is that they need to analyse the whole utterance before they can produce a representation. As has already been stated, there is quite a high likelihood of misrecognition, thereby producing input which is outside the scope of the grammar (ungrammatical input). One option is to constrain the speech recogniser using a finite-state network which mirrors the grammar of the meaning analysis component (section 4.1), thereby forcing the 'speech recognition' component to produce only those sentences which can be analysed using the grammar. An alternative is to dispense with the grammar-based approach altogether and use a caseframe parser.

4.2.3 Caseframe parsers

An alternative view of parsing has emerged as researchers have begun to use natural-language processing in practical applications. Caseframe parsers [17] do not use

an intermediate syntactic stage of processing, but attempt to go straight from the input representation to a representation of its meaning. This is done by assuming that phrases in the sentence have meaning, but only in the context of a predefined frame, or meaning template. An example of a frame is shown in Table 4. As many as possible of the slots of the frame are filled in, using simplistic syntactic analysis, from the phrases of the input sentence. This means that the phrases can appear in the middle of any amount of extraneous information, because only the particular phrases which are relevant to the frame are analysed. This approach is more robust than the grammar-based technique. For example, the following three sentences would be mapped on to the 'spilling' frame as shown.

'the waiter spilled the wine on the couple'

'It is the case, I believe, that earlier last week the waiter, what an appalling employee, spilled some of the wine which we use for only our best customers, on the couple visiting for the first time.'

'when waiter spilled the wine on the couple food'

Table 4 An example of a frame.

Spilling frame	
Actor	waiter
Patient	wine
Location	couple

However, there is a step in the above description that is obviously missing — how to pick the correct frame for the given utterance. This is usually done on the basis of the information contained in the sentence. One method of choosing the correct frame is to allow each piece of information within a sentence to vote for a particular frame. The frame used is the one which has the most votes for the given sentence.

A circular process can develop, however, where only the required information to fill the frame is analysed, but all the information needs to be analysed to pick the right frame. For this reason, caseframe parsers are generally only used in limited domains, where only a small number of frames are considered.

Caseframe parsers are not capable of the fine detail which a good grammar-parser can produce. This may be unnecessary in most applications, but could lead to confusion if the kind of input encountered really can only be distinguished using detailed analysis.

4.2.4 A hybrid approach

If grammar-based parsers are good at producing detailed analyses but poor at handling ill-formed input, while caseframe parsers are good with ill-formed input but unable to produce highly detailed analyses, then a combination of the two would seem ideal. Stallard and Bobrow [18] propose a two-stage process, combining a grammar-based parser followed by a caseframe parser, using a chart [19] as the intermediate representation.

Charts are used by a particular kind of grammar-based parser called a chart parser (this is something of a misnomer, since it is parsing words, not charts). This kind of parser produces all possible analyses of the text, storing intermediate results in a chart. This means that if there is no known analysis for the whole of the input, then there is the potential to look in the chart for incomplete analyses, which will represent those parts of the sentence which can be analysed by the grammar-based parser.

A nice way of combining these fragments is to build them into a frame. A caseframe parser can then be used to extract the meaning of the input utterance, not from the words, but from the chart produced in the first stage. This frame will then provide a meaning representation with as much detail as the grammar will allow, thus improving on the standard performance of caseframe parsers.

4.3 Dialogue manager

The dialogue manager (DM) is the central controller for an SLS. On receiving input from the speech recogniser, the DM manages and co-ordinates calls to the rest of the components in the system (see Fig 2).

The DM knows about the generic structure of conversations. It uses a conversational modeller to build up a dynamic model of the conversation between a user and the system as it progresses. It models conversational turns from both participants matching questions with their answers, instructions with acknowledgements, and so on. The syntactic and semantic analysis of individual utterances (or sentences) produced by the 'meaning extraction' component is used as a base and the DM goes beyond these single-sentence analyses to produce a dialogue model, which could be thought of as a grammar for the dialogue.

4.3.1 System-driven, user-driven, and mixed-initiative dialogues

Conversations with existing IVR dialogue systems, such as telephone banking systems, can be frustrating for callers because the conversation must always follow a strict pattern. A caller cannot say anything he or she wants, but is forced to give the correct reply at the appropriate time,

otherwise the interaction cannot proceed. In this kind of dialogue, the system takes all the initiative and callers are not allowed to ask questions, or say anything which is not specifically asked for. In other words, the dialogue is entirely system-driven.

At the other end of the scale are user-driven systems where the user takes all the initiative. The interface to the BT Business Catalogue (see section 3.1), operates in this way. The system accesses the World Wide Web (WWW) and waits until the user inputs a query.

Dialogue managers are now being developed that allow more natural conversations to take place between humans and machines. The intention is to produce a two-way flow of communication where the initiative can be taken either by a user, or by the system, and information may be given, asked for, and received by either party. Here is an example mixed-initiative conversation with an imaginary movie database query system of the future:

User:	I'd like to see a film starring Meryl Streep.
System:	There are quite a few of them. Are you interested in thrillers, comedies or historical dramas?
User:	What about that one where she plays a Danish farmer.
System:	Do you mean 'Out of Africa'?
User:	Yes, that's the one.

The user instructs the system to find it a film starring Meryl Streep (user-initiative), but the system finds there is more than one and decides to ask the user what category of film (system-initiative). The user chooses not to answer this question directly, e.g. 'comedies', but gives instead some more information about a particular film (user-initiative). Now the system has enough information to pinpoint the film and the user confirms the system is correct.

4.3.2 Ellipsis and anaphoric reference resolution

Speakers use words or phrases to refer to things in the real world. Anaphoric references are a specific type of referring expression and are used when a speaker refers back to something mentioned earlier, e.g. 'he', 'him', 'her', 'that one', 'that way', 'the house' can all be anaphoric reference expressions:

User:	Are there any messages from Peter?
System:	You have two messages from Peter Wyard.
User:	Read his second one.

In the above exchange, the user's second utterance contains two anaphoric references: 'his' and 'second one'. Assuming a grammar-based parser, it will represent 'Read his second one' as an ELF (see section 4.2). The ELF below shows that there is something salient and singular (A) which is to be read, it is the second one, and it is from a masculine person of unknown name:

```
ELF:
    read(A),
   [salient(A),singular(A)],
   [ord(A,2),named(B,C),gender(B,masculine),from(A,B)]
```

It is the reference resolution process which fills in missing information in the ELF. It does this by searching back through the preceding conversation to find something which can be read, and a masculine person. It is assumed that the last masculine person mentioned will be referred to as 'his', although this might not always be the case. The ELF output by the meaning extraction component is thus converted into a resolved extended logical form (RELF) where the missing information (message(A) and name (B,peter)) is filled in:

```
RELF:
    read(A),
   [salient(A),singular(A)],
   [message(A),from(A,B),ord(A,2),name(B,peter),
    gender(B,masculine)]
```

Ellipses occur when something is left out of an utterance which can be determined from what has gone before:

User:	Do I have any messages from Peter?
System:	You have two messages from Peter Wyard.
User:	And David?

Here the user has left out: 'Do I have any messages from..' and has simply said: 'And David?' The 'meaning extraction' component will produce an extended logical form:

```
ELF:
    A,
    [],
   [name(B,david)]
```

and the RELF will include the missing information which was found in the previous logical form:

```
RELF:
    list(A),
   [],
   [message(A),from(A,B),name(B,david),
    gender(B,masculine)]
```

4.3.3 Co-operative responses

Much of the ground work on the nature of conversation has been carried out by philosophers of language such as Grice. Grice devised a set of maxims, or co-operative principles, to which he considered people generally

conform when participating in a conversation [20]. Four of the most important maxims are of quality (be truthful and avoid statements for which you have too little evidence), quantity (be as informative as necessary, but do not give too much information), relation (do not give irrelevant information), and manner (do not present information in a way that is obscure, ambiguous or too lengthy). People tend to be co-operative in conversation for many reasons, for instance:

- to make the conversation flow more easily,
- to assist understanding,
- to pass information efficiently avoiding irrelevant information, false information, or too much detail,
- to promote good social relations.

If the maxims were ignored, then conversation would become very difficult. Of course there are many examples where the maxims are deliberately broken for the sake of humour, pathos or sarcasm. At present, these are outside the scope of this work and, unfortunately, the DMs do not have a sense of humour!

The DM tries to be as co-operative as it can in responding to the user — it tries to obey at least the maxims of quality, quantity and manner. It does this by being as brief in its responses as it can, by ensuring the database is accurate and up-to-date, and by supplying helpful information. The maxim of relation can be harder to satisfy in that the DM cannot always be sure that the information it is giving is maximally relevant to the user. For instance, if the database query finds nothing, then rather than just saying 'none found', the DM attempts to give some information which the user might find helpful. For instance:

User:	Do I have any e-mails from Anna about aardvarks?
System:	Unfortunately you have no matching message, however, you have one message about aardvarks.
User:	OK, but do I have any e-mails at all from Anna?

Here there are no e-mails from Anna about aardvarks, but the system finds a message which is not from Anna but is about aardvarks. The DM does this by generalising the query it is making to the database. In order to do so it must decide which parts of the user's specification are the most important. Is it most important that the e-mail is from Anna or is it most important that the message is about aardvarks? Or are the two equally important? Here it wrongly assumes that the user is primarily interested in aardvarks. Our DM at present works from a priority list compiled from intuition about what might be most important to a user. The whole area of co-operation is one which we intend to investigate further in the future.

When the database query is generalised with a successful result, the DM produces another logical form, the co-operative extended logical form (CoopELF). For the above example, the RELF and CoopELF are as follows:

RELF:

```
list(A),
[plural(A)]
[message(A),from(A,B),name(B,anna),
 gender(B,feminine), about(A,aardvarks)]
```

CoopELF:

```
list(A),
[plural(A)] [message(A),about(A,aardvarks)]
```

Both logical forms are then sent to the response module (see section 4.5).

4.3.4 Error recovery

There are several reasons why a system may not understand a user's request -- speech recognition errors, meaning extraction errors, and conversational modelling errors.

Unfortunately speech recognisers are not 100% accurate and a user does not always use phrases the computer can understand. It can be frustrating for a user if the system simply keeps repeating 'Sorry I didn't understand that'. Sometimes asking a user to repeat an utterance will result in a successful recognition, if, for example, a quiet utterance or background noise resulted in a recognition error. However, sometimes the user will need to rephrase the question, or speak later, or earlier, etc. It is necessary to have an error recovery dialogue which helps the user to try different strategies when difficulties occur, and which seems helpful rather than repetitive.

Sometimes conversational modelling errors occur; for instance, when a user says 'Read her e-mail' and the DM has no record of a female person being mentioned previously, the DM must then ask the user an appropriate question, e.g. 'Who do you mean?'

4.4 Database query

When the DM has prepared the query, it will be passed to the database query component. The database query component's purpose is to convert the query from the DM into one or more queries which can be used to find the required information from within the database. Having established the queries, the database query component then extracts the actual information from the database.

This is not as straightforward as it sounds and may involve a number of steps.

- Using general and domain-specific knowledge to attempt to satisfy the query — an example of this is the use of class taxonomies. If a user asked about 'reptiles', but the database knew about 'snakes', then the database query component could look down a reptiles taxonomy to realise that there is a link between query and data. Similarly, a user asking about 'turtles' from a database of 'terrapins' could have the query satisfied by the database query component if it looked sideways in a reptiles taxonomy.

- Optimising the query — by carefully arranging the order of execution of subqueries with a query, the speed of getting a result can be improved dramatically. To do this, the database query component will need to rank the severity of each of the constraints in the query. The query will execute optimally when the most severe constraints are applied first.

Of course, neither of these operations ensure that there will be a result from the query, i.e. there may be no answer to the query. Therefore the database query component will pass as much information to the dialogue manager as possible about a query which failed, so that the DM can decide what to do next.

Ultimately, the database querying module provides a means of separating the actual database query (in SQL, for example) from the internal representation in the DM. This should mean, in theory, that the dialogue manager need not be aware of the database used, and that using a different database management system only involves changing the internal mapping used by the database query component.

4.5 Response generation

In spoken language systems, responses are normally thought of as being provided using speech. However, in the case of multimodal systems, the response can also be in a graphical or pictorial form. This section is therefore divided into three parts -- the generation of text, the synthesis of speech, and the generation of graphics.

4.5.1 Text generation

Text generation is the task of putting into words the information that is to be sent to the user. It can be as trivial or as complex a process as is appropriate for any particular SLS.

At the trivial end of the scale, text generation can be avoided altogether by using canned-text sentences. These work well if the application is very simple and there are only a very limited number of things the system will need to say.

The next step up from this is to have carrier or template sentences where relevant information can be slotted. This allows a little more scope for personalising the messages to the user and has been used successfully in ELIZA [21], the famous artificial intelligence program which attempts to fool people into believing they are communicating with a psychologist. ELIZA recognises and scores certain patterns in the user's input and uses the highest scoring pattern as a filler in its template output, transposing personal pronouns and possessives such as 'I' for 'you', and 'my' for 'your'. For instance:

User: I am worried about my mother.
ELIZA: Tell me more about your mother.

The template output sentence is 'Tell me more about...' and ELIZA has transposed 'my' for 'your' in the pattern 'my mother' to give the filler 'your mother'.

The most flexible method is to generate the most appropriate responses as and when necessary. Just as grammars can be used for analysis, most can equally well be used for generation. However, the flexibility of a text generator depends on the size of the grammar and lexicon used. If they are very small, the effect is little better than using template sentences.

Text generation is the reverse process of parsing. A parser converts a string of text to some kind of knowledge representation, whereas a text generator converts a knowledge representation to a string of text. For the e-mail assistant SLS, the RELF which has been instantiated after a successful database query is used for generation. If the initial database query was unsuccessful, then the uninstantiated RELF and the CoopELF are both used for generation. Furthermore, the presuppositions part of the RELF is used to reply in an appropriate way depending on the assumptions made by the user. Thus if the user was expecting more than one e-mail as in 'Read me my messages from Peter' and only one exists, then the system can reply 'There is only one, the message reads:...'

4.5.2 Graphics generation

Multimodal systems are enhanced by graphs and/or pictures incorporated into the text. A graph or picture can show at a glance what would often take many paragraphs to describe in words.

BusCat (see section 3.1) builds WWW pages on-the-fly in order to answer a user's query. The WWW pages include pictures of BT products relevant to the user.

A consideration when (automatically) collating material to be shown on a computer screen is size of images and graphs. It is desirable to try to fit the information within the limits of the screen rather than forcing the user to scroll down to view the whole page. The interface to the BT Business Catalogue uses different sized images to attempt to achieve this.

4.5.3 Speech synthesis or text-to-speech

BT's SLS systems all use the Laureate speech synthesiser [2] whenever TTS is required. When a text for synthesis is being generated automatically by the response module, information about the structure of the text will already be known. This opens up the possibility of providing Laureate with details of what kind of utterance it is (declarative, interrogative, or imperative), where the pauses should be, where the strongest emphasis ought to be placed, and so on. This is an area for future research.

5. Futures

There is a considerable amount of work to be done in taking the demonstration systems described in section 3 and turning them into applications that can be used by customers. To a certain extent this is the normal process of downstreaming, addressing issues such as thorough coverage of the application domain, robustness of software in the hands of naive users, and a well-designed user interface. However, there is further work to be done on underlying technology to make continuous spoken-language systems which will be really effective. Three such technology areas are highlighted in this final section. These technologies are aimed at improving the quality of the system by making it more user-friendly and more intelligent.

5.1 Speech recognition

SLSs of the kind described in this paper place great demands on the recognition component, and these demands will increase as the coverage of the system is extended, because the recogniser must still work with sufficient speed and accuracy with a larger vocabulary and higher perplexity of language. In addition to these fundamental requirements, there are some other features which would help improve performance.

- Spontaneous speech recognition — this involves a number of different areas, each of them trying to enable better recognition of speech from naive users in a real application situation, as opposed to experts running a controlled demonstration. Non-speech sounds, such as breath noises, hesitations, coughs, etc, must be accommodated, together with out-of-vocabulary words (because it will never be possible to create a recogniser vocabulary which covers every word to be spoken by real users), and the disfluencies, hesitations and restarts of spoken language.

- Confidence measures — the recogniser should be able to give the dialogue manager an idea of how confident it is that it has recognised an input utterance correctly, and beyond this, its confidence in particular parts of the recognised string. For example, the recogniser might say that it has recognised 'which films are UNK by Clint Eastwood', in which it is highly confident about everything except the unknown word UNK.

- Speaker adaptation — in many cases it is anticipated that SLSs will be used regularly by the same individual, in which case they may be willing to train a speaker-adaptive system to their voice before using the system. If this is not possible, the dialogues with SLS are usually long enough to make speaker adaptation during the dialogue a useful proposition.

- LM adaptation — it is important that the recogniser can efficiently update its language model during the course of the dialogue, as explained in section 3.1.

- Prosodic information — in the context of a spoken dialogue, there is considerable prosodic information in the user's utterances (stress, intonation, etc) which could help the recogniser, but is currently ignored.

5.2 User-centred language

The ideal would be for users to be able to speak to an SLS in a language which is natural to them, although it is realised that people will be prepared to adapt to the system to a certain extent. To do this the way real users actually say things must be studied, and incorporated as far as possible into the recogniser and grammar. However, user-centred language also requires more sophisticated processing in analysing the input sentence and translating it to a suitable database query. To take a very ambitious example, a user might say: 'How can we cut the cost of getting bulky documents from Edinburgh to London?', which would ideally be translated into a database query about faxes and ISDNs. This kind of translation requires considerable knowledge bases in the system, both domain-specific and general world knowledge, and it also requires a powerful inference engine to make effective use of the knowledge bases. In the shorter term, such queries may be beyond our capabilities, but users will not expect to have to phrase everything in exactly the terms encoded in the database which contains the required information.

5.3 More flexible and intelligent dialogues

The dialogues discussed are already considerably more flexible than the traditional prompt and response dialogues. However, there are two areas where dialogues could be enhanced:

- greater use of mixed-initiative dialogues, where the system sometimes takes the initiative -- for instance to give the user some background information it has inferred they may be unaware of, or to make a 'sales pitch' for a particular product or service, if, for example, the user has been spending a lot of time asking questions about it,

- more use of user and life-style profiles -- these are knowledge bases which the system has built up about the particular user, and the category to which the user belongs, the dialogue manager then being able to use this information both to help it interpret user input, and to decide on suitable initiatives to take, as discussed above.

6. Conclusion

Spoken language systems are likely to be of great importance in the future development of computer technology and its adoption by large numbers of people in the information age. this is because language is one of our primary means of communicating with each other, and unless humans change fundamentally in a a short time, the use of natural language will be an important element in facilitating communication between humans and computers.

This paper has discussed the components of spoken language systems in general terms, and described three systems which are under development at BT Laboratories. There are a variety of spoken language systems, from pure speech-in/speech-out systems to multimodal systems which aim to combine spoken language with other modalities, such as typed text and mouse clicks, in order to achieve the most user-friendly interface possible.

As for the future of spoken language systems, the current state of the art is that they are on the verge of commercial deployment in some domains. However, it will be some time before we approach the scenarios envisaged in '2001: A Space Odyssey' or 'Star Trek' where one can talk to a computer system in a completely natural way.

References

1 Scahill F et al: 'Speech recognition -- making it work for real', BT Technol J, 14, No 1, pp 151—164 (January 1996).

2 Edgington M E et al: 'Overview of current text-to-speech techniques: Parts I and II', BT Technol J, 14, No 1, pp 68—99 (January 1996).

3 Bissell R A and Eales A: 'The set-top box for interactive services', BT Technol J, 13, No 4, pp 66—77 (October 1995).

4 http://www.cm.cf.ac.uk.movies

5 Power K J: 'The listening telephone — automating speech recognition over the PSTN', BT Technol J, 14, No 1, pp 112—126 (January 1996).

6 Gazdar G and Mellish C: 'Natural language processing in Prolog', Wokingham, England, Addison-Wesley, pp 169—174 (1989).

7 Rayner M and Wyard P: 'Robust parsing of n-best speech hypothesis lists using a general grammar-based language model', in 4th Eurospeech Conference on Speech Communication and Technology, EUROSPEECH '95, pp 1793—1796 (1995).

8 Special Issue on 'Ill-formed input', American Journal of Computational Linguistics (1983).

9 Hodges W: 'Logic', Hamondsworth, England, Penguin Books (1977).

10 Warren D H D and Pereira F: 'An efficient, easily adaptable system for interpreting natural language queries', American Journal of Computational Linguistics, 8, No 3—4, pp 100—119 (1982).

11 Fillmore C: 'The case for case', in 'Universals in Linguistic Theory', New York, Holt, Rinehart and Winston, pp 1—90 (1968).

12 Beardon C, Lumsden D and Holmes G: 'Natural language and computational linguistics', Chichester, England, Ellis Horwood, pp 150—156 (1991).

13 Allen J: 'Natural language understanding', The Benjamin/Cummings Publishing Company, p 41 (1987).

14 Chomsky N: 'Syntactic structures', The Hague, Mouton Publishers, p 15, (1957).

15 Cann R: 'Formal semantics', Cambridge, England, Cambridge University Press, p 5 (1993).

16 Alshawi H (Ed): 'The core language engine', London, England, MIT Press (1992).

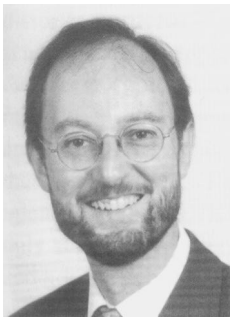
17 Carbonell J G and Hayes P J: 'Robust parsing using multiple construction-specific strategies', in Bolc L (Ed): 'Natural language parsing systems', Springer-Verlag, Berlin, pp 1—32 (1987).

18 Stallard D and Bobrow R: 'Fragment processing in the DELPHI system', Proc of Speech and Natural Language Workshop, DARPA, pp 305—310 (1992).

19 Earley J: 'An efficient context-free parsing algorithm', Communications of the ACM, 13, No 2, pp 94—102 (1970).

20 Grice H P: 'Logic and conversation', in Cole P and Morgan J L (Eds): 'Syntax and Semantics', 3, SpeechActs, pp 41—58, New York, Academic Press (1975).

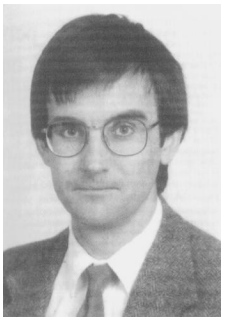
21 Weizenbaum J: 'ELIZA -- A computer program for the study of natural language communication between man and machine', Communications of the ACM, 9, No 1, pp 34—45 (1966).



Peter Wyard obtained an honours degree in theoretical physics from Cambridge University and an MSc in astronomy from the University of Sussex. After ten years teaching physics in India and the UK, he took an MSc in digital systems from Brunel University and joined BT laboratories in 1988. He first worked on connectionist natural language processing. After a period as project leader for speech technology evaluation, he returned to the language group where he now leads the work on developing spoken language systems.

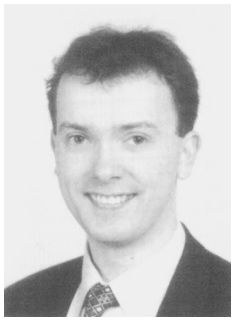
Alison Simons joined BT in 1982 as a sponsored student. After receiving the degree of BSc in Computer Science from the University of Manchester in 1986, she joined BT Laboratories and began work in the speech synthesis research team. She subsequently switched to speech recognition where she led a research team developing BTås first sub-word unit speech recognisers.

She currently leads a team which is developing spoken language systems for accessing information sources and has recently completed the BT MSc in Telecommunications Engineering.



Steve Appleby joined BT in 1983 to investigate methods for locating buried plant after gaining an honours degree in Physics with Musical Acoustics from the University of Surrey. After working in the copper access group on various problems related to copper network maintenance he joined the Systems Research Division to work on a wide variety of topics including fractal population modelling and mobile agents.

Currently he works in the natural language group on semantic aspects of natural language processing and is completing a PhD in fractal population modelling.



Ed Kaneen began work for BT in 1990 as a sponsored student, and joined the language group full-time in 1994, having gained an MEng in Computer Systems and Software Engineering from the University of York. He worked initially on prosody for text to speech, before returning to the work of his MEng, on the parsing of ungrammatical language. This has continued with the language group, focusing particularly on the interface with speech recognition.

Sandra Williams' early career was a varied and included seismic survey analysis, customer relation work, and stained glass craftwork. She joined BT Laboratories in 1988 after graduating from Sussex University with a BA degree in Artificial Intelligence. She joined the natural language group to work on automatic text summarisation.

In 1992, she obtained an MPhil degree in Computer Speech and Language Processing from Cambridge University, which was sponsored by BT. Since 1992, she has been part of the language group in Systems Research. Her specialist work interests include automatic text summarisation, and discourse and dialogue analysis. She is developing dialogue managers to handle ever more natural conversations between humans and computer.

Keith Preston gained a BSc(Hons) in Applied Physics and Electronics from the University of Durham in 1978. After joining BT he was involved in pioneering work in optical communications, and has many publications and patents in this area. He was also responsible for the design and prototyping of a commercial range of advanced laser transmitter products. In 1992 he moved into the area of advanced software and now heads the natural language group at BT Laboratories.