# WP6: Dialogue Manager Functional Specification

Eric Bilange, Norman Fraser,
Nigel Gilbert, Marc Guyomard,
Paul Heisterkamp,
Scott McGlashan, Jacques Siroux,
Jutta Unglaub, Robin Wooffitt
and Nick Youd

June 1990

|  |  |  |
|---|---|---|
| **TITLE** | : | WP6: Dialogue Manager Functional Specification |
| **PROJECT** | : | P2218 - SUNDIAL |
| **AUTHOR** | : | Eric Bilange, Norman Fraser, Nigel Gilbert, Marc Guyomard, Paul Heisterkamp, Scott McGlashan, Jacques Siroux, Jutta Unglaub, Robin Wooffitt and Nick Youd |
| **ESTABLISHMENT** | : | Cap Gemini Innovation, Daimler-Benz Forschunginstitut, IRISA, Logica Cambridge Ltd, University of Erlangen, University of Surrey |
| **ISSUE DATE** | : | June 1990 |
| **DOCUMENT ID** | : |  |
| **VERSION** | : | 1 |
| **STATUS** | : | Final |
| **WP NUMBER** | : | 6 |
| **LOCATION** | : | Mikhail |
| **KEYWORDS** | : | Deliverable, Dialogue, Specification |

# Abstract

This report is the first deliverable from Work Package 6: Dialogue Management. It consists of a functional specification of the dialogue manager to be incorporated into the intermediate Sundial demonstrator system. Chapters of the document provide a motivation for including a dialogue manager in a speech understanding system, define the subset of dialogues with which the Sundial system is concerned, that is, *information providing dialogues*, and describe a range of phenomena to be found in these dialogues. Further chapters consider approaches to handling these phenomena, propose an architecture for the dialogue manager and describe the interfaces between the dialogue manager and two other parts of the Sundial system, the linguistic processor and the message generator.

# Contents

# Chapter 1

# Introduction

## 1.1 Overview of the specification

The objective of the SUNDIAL project is to design and implement prototype computer systems which will interact with users over telephone lines to answer queries within restricted task domains such as hotel booking, flight enquiries, flight booking and train enquiries.

This document specifies the functionality of a 'dialogue manager' module to be included in the first 'intermediate demonstration' SUNDIAL system. This demonstrator is scheduled to be completed in July 1991. The document is the first deliverable from Work Package 6. The first working implementation of the dialogue manager itself will be the second deliverable.

As well as giving an outline of the subsequent chapters and motivating the need for a dialogue manager in the Sundial system, **chapter one** includes a section on the terminology used in the subsequent chapters.

Three different applications have been chosen for the SUNDIAL system. These are *flight enquiry* and *reservation*, *hotel reservation* and *train time table enquiry*. By showing their common general structure, **chapter two** demonstrates that they are all instances of *Information Providing Dialogues*, the application basis for the common system to be developed.

In **chapter three**, some of the dialogue phenomena which have been observed in information providing dialogues are described and classified. These dialogue phenomena have been culled from different corpora: some human–human, and others human–machine. The latter were obtained through simulations, in which callers believed they were speaking to a computer speech understanding system although in fact they were conversing with a person, or a computer under direct experimenter control. The aim of this chapter is to *describe* what people do in dialogue with each other rather than to *prescribe* what the SUNDIAL system ought to be capable of.

First, the chapter deals with the overall organization of dialogues by introducing the concept of 'dialogue phases'. Second, local structure is described at the level of exchanges or utterances. This refers to phenomena such as 'adjacency pairs', 'turn taking' and 'failure–repair'. The following section introduces phenomena tied to informational structures, such as 'reference', 'focus' and 'ellipsis'. The final section is dedicated to prosodic phenomena in dialogue.

**Chapter four** considers a number of approaches to handling the dialogue phenomena described in chapter three within a speech understanding system. It first presents a section on the 'state of the art'. The major types of phenomena are considered individually and their implications for the functionality of the dialogue manager are drawn out in the next section. A number of arguments for the inclusion of a user model into the dialogue architecture are discussed. Knowledge representation is an important issue in any natural language processing system, and the next section first emphasizes the need for declarative representations for facilitating portability across languages and application domains. The different types of knowledge which need to be stored, updated and kept accessible by the dialogue manager are described.

The next chapter, **chapter five**, begins with a section on 'priorities'. This indicates which of all the phenomena described in Chapter 3 are likely to be the most important to handle and which of the approaches described in Chapter 4 are both technically feasible and to be recommended for incorporation in the Sundial intermediate demonstrator.

The chapter continues with a description of the proposed functional architecture of the Dialogue Manager in terms of submodules. Their interfaces, objectives, scope and principles are specified as well as the possible communications between each of these modules.

**Chapter six** describes the Dialogue Manager's interfaces to the linguistic processing and message generation modules. It is intended that all messages passed through these interfaces are expressed in a common interface language. The elements of this language are explained and the functionality of the dialogue manager sub-modules which manage the interfaces are described.

## 1.2 Motivation for the Dialogue Manager

One distinctive feature of the SUNDIAL system is the use of a ‘dialogue manager’ to organize the oral interaction, or dialogue, between system and user. Many previous information service systems did not include such a module in their design and thereby lacked a principled means of structuring the interaction between system and user. Such systems were based upon the design diagrammed in figure **??**

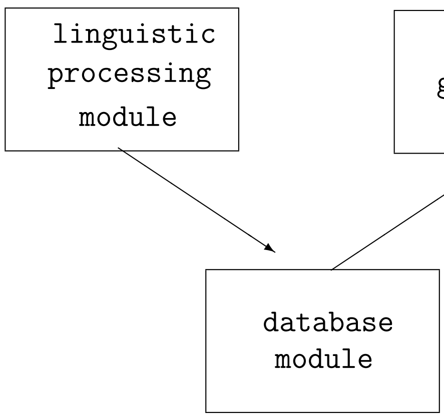

Figure 1.1: A Simple Information Service System

The system consists of three major functional modules: a linguistic processing module, a message generation module and a database. The function of the linguistic processing module is to take an analogue signal from the telephone line and process it, via a series of sub-modules, so as to output a semantic representation of the user’s query. This query is then passed to a database module which attempts to answer the query. The database answer would then pass to a message generator which, again via a series of sub-modules, translates it into synthesized speech output.

The problems that arise from the design of this simple information system provide a rationale for the incorporation of a dialogue manager within the SUNDIAL system. The general problem is that without some means of managing the interaction between the system and the user, the resulting interaction is ‘unnatural’ from the user’s point of view— the system does not conduct a dialogue with the user in the same, or similar, way to how people conduct information service conversations with each other. This ‘unnaturalness’ is evidenced by the lack of interactional properties such as the following:-

- **Extended Interaction**. The largest unit of interaction is frequently a single question answer pair: the user asks a question and the system gives an answer. Naturally occurring dialogues, on the other hand, are extended across more than one sand these dialogues display a sequential organization. For example, dialogues consist of an opening sequence, the body of the dialogue and a closing sequence.

- **Indexicality** Utterances are interpreted (and produced) in a 'context-independent' manner. In natural dialogue interpretation is 'context-dependent': participants oriented to the content and structure of previous utterances, or past context, in order to determine the (situated) significance of the current utterance. Ellipsis, for example, is commonplace in natural dialogue since participants make use of past context in order to interprete 'incomplete' utterances.

- **Mixed Initiative** The initiative is the sole responsibility of one participant. In natural dialogue, either participant can take the initiative. For example, an agent may initiate a clarification of the user's request.

- **Co-operative Reponses** If the system fails to find the information requested by the user, a simple negative response is generated. In natural dialogues, co-operative responses such as suggestive answers may be given in such circumstances.

- **Repair Strategies** If the system is unable to interprete an acoustic sequence or fails to understand an utterance, the result is a simple error message. In natural dialogues, one particpant may correct the problem so allowing the dialogue to the dialogue can continue.

Incorporating this sort of functionality within an information service system is a manifestation of a user-centred design strategy: rather than simply design the system from point of view of information *exchange*, the functionality of the system is increased so as to give an *interaction* between system and user, an interaction which displays properties of natural conversation. The main purpose of the dialogue manager then is to provide to provide the system with just the sort of functionality so that the resulting dialogue is natural, comfortable, comprehensive and comprehensible for the user. In terms of the architecture of the system, the dialogue manager is a module which manages a dialogic interaction between system and user by controlling the interaction between the linguistic processing module, the message generation module and database module. With this sort of approach, the system should be able to 'repair' fragmentary representations produced the linguistic processing module, provide a 'dialogue history' with which the message generator generates elliptical utterances, and so on. The overall architecture of the SUNDIAL system is shown in figure **??**.

In figure **??**, the Dialogue Manager is shown as a single software module that co-operates with other modules. The overall task of the Dialogue Manager can

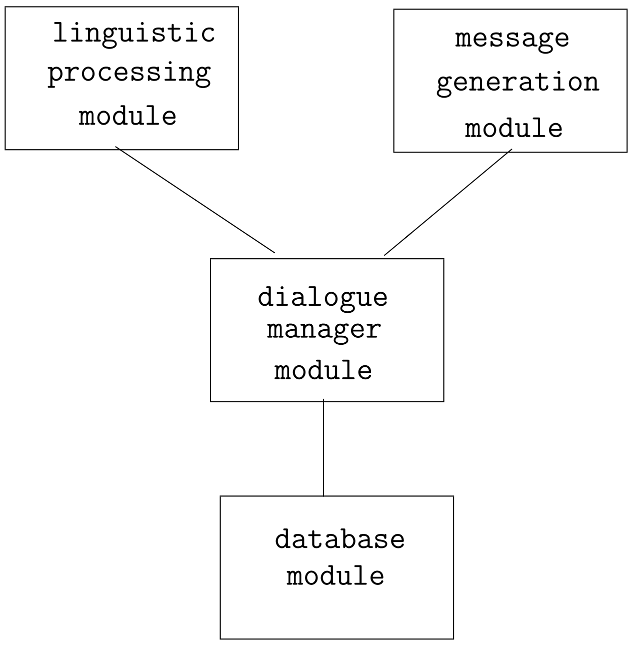

Figure 1.2: The SUNDIAL Information Service System

be divided into several subtasks and for the sake of modularity each of these subtasks can be assigned to a sub-module of the dialogue manager.

Although in an information providing system the main 'speech acts' seem to be 'question' and 'answer', generally a dialogue is not restricted to question–answer pairs. Cooperative communication partners will use many control acts (metacommunicative utterances), to seek confirmation that they have heard or understood correctly, e.g. "Did I get that right?", and to clarify something, e.g. "Sorry, what do you mean?". Furthermore, general conversational principles require the exchange of certain acts of politeness, like greeting formulas and those of expressing one's thanks (see [**?**]).

One approach to understanding dialogue suggests that, based on hypotheses about the knowledge and intentions of the dialogue partner, speakers use certain interactional strategies established by social convention in order to reach their communicative goals. Such strategies are based on the knowledge of typical patterns of communicative acts for different types of dialogues.

If this is the case, it should be possible to determine patterns which represent typical sequences of individual communicative acts for task–oriented, information providing dialogues. It would only be by sharing such patterns that dialogue partners would be able to interpret the communicative function of utterances correctly and to develop an adequate strategy to reach their goals.

Generally, two kinds of communicative acts may be distinguished, 'control acts', which are used to manage the course of the dialogue, e.g. performative introductions of questions and greeting formulas, and 'informative acts', which lead to changing the state of knowledge of the partners, e.g. the utterance "The train leaves at five p.m.".

Apart from the dialogue context an utterance takes place in linguistically, there are also various situational factors which make up or determine the meaning of an utterance. The communicative situation includes the general social relationship among the partners, i.e. matters of social hierarchy, authority, etc., knowledge of the specific communicative setting they are acting in and general world knowledge. The dialogue context can also include (pre–)suppositions about the state of the partner's knowledge derived from discourse objects which have already been mentioned explicitly or implicitly during the dialogue.When using pronouns, definite articles or ellipsis, a speaker is referring to knowledge (s)he assumes the hearer to know.

The main task of the dialogue module, then, is to represent these contextual aspects and the structure of the dialogue, as well as the communicative function of utterances.

According to [?] the following groups of subtasks can be distinguished in representing the communicative competence of the system:

- First, there are tasks related to the process of understanding, like resolving semantic and pragmatic ellipsis and identifying the discourse objects referred to. Recognizing and identifying communicative acts, based on expectations resulting from the dialogue context, as well as assessing alternative interpretations relative to the context and deciding on one of them, also belong to this group.
- Secondly, there are tasks related to dialogue structure, like planning the course of the dialogue in order to reach one's communicative goals according to an adequate strategic plan. Depending on the type of communicative act, different (re–)actions are to be taken. If the system is to be active, an answer or a question has to be uttered, i.e. the system has to perform an adequate communicative act.
- Third, there are cognitive tasks resulting from the understanding process to be performed. If possible, an answer has to be generated, and, if not, a question to the user needs to be constructed. This will be determined by the current topic of the dialogue. Furthermore, the dialogue history has to be updated with the form, content and communicative function of the last

utterances. Predictions about the next communicative act, topic, and — if possible — the wording to be expected next from the user need to be made.

As described in chapters five and six, declarative knowledge bases are needed for the system, in order for the dialogue manager to be able to perform its communicative tasks. There should be a dialogue model representing the patterns of different sequences of communicative acts in information–providing dialogues, independent from the specific application domains.

The dialogue history should store the discourse objects mentioned in the dialogue together with their syntactic and semantic structures, particularly those which can be referred to in the subsequent course of the dialogue. Predictions for subsequent utterances which can be deduced from those already mentioned can be stored in the dialogue history as well.

The following chapters describe these requirements in more detail, together with proposals for their implementation in the intermediate demonstrator. First, however, it is necessary to define some basic terminology.

## 1.3 Terminology

In a spoken dialogue, the physical data are acoustic events, i. e. sound waves emitted by the participants. During the analysis of this signal, units are detected and combined to form other units. As long as this combination proceeds from smaller to larger units, the layers of analysis are quite distinct. Phonemes are detected, combined to word units and these are used to give one or more (in the case of more than one phrase) syntactic and/or semantic representation, first of constituents and then of whole phrases and sentences. Though there may be minor problems as to which kind of unit belongs to which part of the analysis, this is the classical distiction between phonetics, (morpho-)syntactics and semantics, and it is quite straightforward.

Now, what happens when one and the same sound event has to be seen as a unit (or several units) in more than one system of reference, i.e. in the context of more than one analytical framework? Of course, each of the frameworks involved must be recognizable by the way it addresses its units. So, to avoid confusion, it is important to keep the analytical frameworks apart and to assign special terms to each unit. Furthermore, if the terminology is not clear, there is a tendency for misunderstandings to remain undetected. Though terminology always has theoretical implications, both in connoting certain theoretical backgrounds and in favoring certain views of things (especially if it is largely metaphoric), it is possible to identify certain units and relations between them without commiting oneself too much to one theory.

The aim of this section is to separate strata of analysis clearly and distinctly. It is not concerned with the systems of combining these units, such as e.g. dialogue stucture on the basis of communicative function.

For example, some describe dialogue structure in terms like "turn", "turn-taking", "turn-giving" etc. Others try to describe them by means of dialogue control functions (see below).[1]

To illustrate the following, we use an example dialogue, given in the German original and an English translation. This example is taken from the FACID-Corpus ([**?**] :dberl11). It is a human-human information dialogue via the telephone. The subject is train-timetable information on a train from Hanover to Nuremberg. The transcription is the "Halbinterpretative Arbeitstranskription" (HIAT) (Semi-interpretative working transcription) [**?**, **?**]. S(peaker)1 is the subject seeking an information, S(peaker)2 is the information service. The index ($S1_X$) is a counter of the line number for each speaker. Every horizontal line represents approximately 10 seconds duration.

### 1.3.1 Utterance

We will call one continuous sound signal from one speaker an **utterance**. An utterance will mostly be speech, but it may also consist of paralinguistic phenomena, such as sighs, grunts etc. Not rated as an utterrance are such sounds that are made involuntarily, like coughs, sneezes etc. These are rated as noise. We only deal with speech communication via the telephone. Non-verbal communication cannot take place. In the example, *one* utterance is $S2_4$ continued on $S2_6$.

### 1.3.2 Phrase

Every utterance consists of words, and it has a syntactic level. The words have a certain sequence that can be analyzed as a structure. Given the acceptability range, i.e. the grammatical competence of the grammar, we call a sequence of words that a structure can be assigned to by the parser a **phrase**. A phrase thus does not have to be complete or grammatical in the sense that it is 'correct' from the point of view of prescriptive grammar, nor is it necessary that it is a complete sentence. It may be a single word like "yes" or "no", or it may consist of several sentences. What is important is that the parser was able to

[1]see [**?**]pages148-152 for some arguments against the notion of "turn".

**Example:**

```
S1_1      daß ich noch vor Mitternacht ankomm, nä         ja
S2_2                                       vor     Mitternacht

S1_3
S2_4      ( ... )  des wär’ Abfahrt in Hannover / hm / deen / an welchem

S1_5                               des wär’ der kommende Sonntag
S2_6      Sonntag wär’n des genau ?                              kommende

S1_7
S2_8      Sonntag . des wäär sieb / achtzehn Uhr elf ab Hannover ....
```

---

```
S1_1      so that I arrive before midnight, right         yes
S2_2                                   before     midnight      ( ... )

S1_3
S2_4      that would be departure Hanover / er / thee / which sunday

S1_5                              that would be next sunday
S2_6      exactly would that be ?                           next sunday .

S1_7
S2_8      that would bee sev / eighteen eleven  .......
```

give a structural description, including, as far as possible, the semantics. In the example, the utterance $S2_{4\&6}$ can be analyzed as two phrases. The question starting at “Welcher” is a complete phrase. The first part of the utterance (up to “deen”) is incomplete. The “deen” (transcribed for “den” with a lengthened vowel) was clearly meant to be the beginning of a phrase containing a date, like “den vierzehnten September”, and as such structurally belongs to the first (incomplete) phrase. There is still one problem: the non-word “hm” that separates the “deen” from the first part of the sentence. For the moment, we will leave it out and return to it later.

### 1.3.3 Communicative function and dialogue act

A **dialogue act** is a substructure of an utterance that a pragmatic value has been assigned to. This substructure may be the whole utterance or a part thereof, such as a phrase. A pragmatic value is one of the **communicative functions** (cf. [?], [?], [?]). The actual set of functions is dependent upon the type of the dialogue, in our case information providing dialogue (see Chapter 2). The communicative function of (a part of) an utterance is the function by which the (part of the) utterance changes the dialogue context, i.e. brings about a new state of the dialogue and of the mutual knowledge of the participants. The assignment of a communicative function depends on a set of rules taking into account linguistic clues that are found in the (part of the) utterance, the dialogue history and/or the supposed plans and goals in a partner model as well as the possible continuations of the dialogue. Thus, a dialogue act is the content of a (part of an) utterance set in relation to the possible communicative functions.[2]

There are two basic categories of communicative functions, *viz.* informative and control functions, the assignment of which to utterance-substructures yields two kinds of dialogue acts: Factual informative acts that convey factual information, both on the subject matter of the dialogue and on the state of the plans and goals of a participant, such as “I want to know ...”, and Dialogue control acts that are used to explicitly manage the course of the dialogue. They deal with the communication itself, e.g. clarifications, thanks or confirmations. These acts are not concerned with information transfer. Some of the dialogue control acts may be metacommunicative, i.e. have as their subject parts of the communication, as in a clarification question (“What do you mean by this word?”). However, not all metacommunications are dialogue control acts (the answer to the example question above “By this word I mean ...” is meta-communicative but not a dialogue control act), nor are all dialogue control acts metacommunicative (“Good

[2][?], page 70ff, (a paragraph not contained in the draft version of the paper [?]) points out that the notion of communicative functions and, hence, dialogue acts, is not theoretically indispensible, but rather a convenient way of handling bundles of appropriateness conditions.

morning" is not metacommunicative).

In the example, the first phrase of S2's utterance beginning on line 4 can be assigned the function "INFORM" in Bunt's classification of informative acts in information dialogues. So, this phrase is an information-giving act, corresponding, in this domain, to the "acte REPNUP" in the dialogue act classification of [?]. But this act is not completed. It breaks off in the middle and S2 starts a new dialogue act by asking S1 to supply a missing parameter (cf. the DEMPARxxxx-acts in [?]). After S1 has answered this request, S2 starts the information giving again. But between the two, S2 indicates by "hm", that there are some problems with the continuation of this information-giving act. The communicative function of this part of the utterance is a dialogue control function, namely a problem-indicator. Thus, even though there can be a single structural description of the first phrase, there are two dialogue acts in this phrase. What is important is that phrase and dialogue act do not necessarily fall together. There may be discontinuations and, at least in the same utterance, embedded communicative acts.

The following figure shows the different layers of analysis of the example utterance $S2_{4\&6}$.

```
     COMMUNICATIVE ACT
     +----------------1------+ +2+ +1'+ +---3----------------------+

     PHRASE
     +------------------------------+ +---------------------------+

     UTTERANCE
     +------------------------------------------------------------+

S2   des wär' Abfahrt in Hannover / hm / deen / an welchem Sonntag wär'n des genau ?
t    |---------------------------------------------------------->
```

## 1.3.4 Topic and Focus

Two further terms have to be made explicit. The first one concerns the notion of topic. By this we mean the discourse topic, i.e. the subject matter the dialogue is about. The focus, in the way we use the term, is the actual center of attention. In the example, the topic is the train connection the caller (S1) wants information about. The topic may change, but this has to be made explicit with dialogue control acts like "I have another question". The focus, on the other hand, may change very rapidly. In $S2_6$, for example, S2 put the exact date of the journey into focus by asking for it. After S1 has answered the request, this date is no longer in focus. So, our usage of these terms must not be confused with other usages, like the notion of "topic" as opposed to "comment" in the Prague School or the notion of "prosodic focus" in phonetics.

# Chapter 2

# Information Providing Dialogues

## 2.1 Introduction

In Sundial, three different applications for the intermediate and the final system have been chosen. For the French and British partners, the application is flight inquiry and reservation. For the Italian partners, it is hotel reservation, and for the German partners, the application is train time table inquiry. In this chapter, we will show that all the applications call for dialogues that are instances of *Information Providing Dialogue*, and as such the expected dialogues have a common general structure.

One of the advantages of having a range of different information providing applications is that, as we shall see, it then becomes easier to identify which features are characteristic of Information Providing Dialogues and which are specific to one or other of the applications. It is a requirement for the project that application specific issues are kept distinct and that the remaining generic aspects are clearly identified and interchangeable between applications.

First, we will give an outline of the different application areas. From these we will then abstract the tasks the dialogue has to fulfill. Finally, we will give an outline of the underlying structure of an Information Providing Dialogue.

## 2.2 An outline of the applications

All of the applications in SUNDIAL are intended to be accessed via a telephone line and open for use by the general public. In the following sections, each application is briefly described. Much fuller details about these applications, including an explanation of why they were selected and their particular characteristics, can be found in Chapter 1 of [**?**].

### 2.2.1 Airline timetable inquiry and reservation

The applications chosen by the French and British partners differ only slightly (apart from language-specific issues, of course). That is why they are treated here together. They both offer information about flight connections between single airports, i.e., not about flight connections that require change of aircraft or airline. Furthermore, the French partners want to add a booking facility, though this would only be open to users known to the system, who also have had training in the application area, such as secretaries of companies who regularly book flights for their companies.

Callers (or, in terms of a computer system, users) telephoning the flight information service may have any of several distinct needs. They may want information concerning certain flights, like arrival or departure time, destination etc., or they may want to book one or more seats on a flight. The information system should provide them with the information they need from a database, and, if necessary, make changes to this database, e.g. mark certain seats on a flight as reserved.

### 2.2.2 Intercity train timetable inquiry

The German application differs from the Anglo-French one in several ways. First, the domain is that of train connections by Intercity services. Second, the subject of the information offered is not only about direct connections, but also includes connections through intermediate stations. This means that information about itineraries needs to be given, including stopover places, stations and times at which trains have to be changed etc. On the other hand, a booking facility is not foreseen as part of the system.

Callers can thus ask about train connections between all of the German cities that have stops in the Intercity network. Furthermore, they can also specify the path by which they want to travel, i.e. they can say that they want to change trains at specified stations or that they want a certain one out of several alternative lines. They can also obtain the same data about connections as in the airline information system, *viz.* departure and arrival times etc.

### 2.2.3 Hotel reservation

The Italian partners have chosen as their application a hotel room reservation service. Callers can specify the city, time and price category, and they get offered a choice of hotels with vacancies. In later versions of the system, they will also be able to book rooms.

The service will use an existing database which can currently be accessed through a menu-based interaction on a character terminal. The basic data in the data base are updated every six months by the CEI and by hotels themselves. For each town or city, the database provides information about the region in which it is located, its altitude above sea level, its population, the telephone number of the local tourist information office and its distance from major road and rail junctions. Details held about each hotel include its name, address and telephone number, its location in the town, the dates it is open and the charges for rooms.

The intended information system will allow access to these data either by specifying a town or a region. The caller will then be able to further specify the particular hotel or town desired.

## 2.3 General task structure of the applications

Though all of the applications are in different areas, they have a common structure. From the point of view of the informant, the dialogue has to proceed along the following logical steps:

1. Identify what the caller wants
2. Elicit from the caller the parameters needed to specify adequately the item in question
3. Consult the data
4. Give the desired information, if any
5. (Optionally) take a reservation note from the caller
6. Make the reservation known to others, i.e. update the database

Step (1) includes testing whether the information can be provided in the first place. The information provider has to know what s/he has information about and what not, about what s/he can do or look up and what s/he can't. Clearly, it is very difficult to incorporate into a computer system (meta-)knowledge about what it doesn't know[1], although it is easy enough for the system to notice that it cannot answer a user's question. As noted in Chapter 3, such difficulties will be dealt with in SUNDIAL using techniques to detect failure and offer repair.

Steps (2) and (3) are linked by the term 'adequately'. The information provider has to check the data to see whether the identification of the item in question has been narrowed enough for him/her to give the desired information. This can be the case after the caller's first utterance, which may contain all the parameters needed. But, normally, further parameters have to be requested by the information provider singly or in groups, according to which parameters gives the best approximation for the identification. For example, for train inquiries, departure and destination place are usually the most important, as they permit checking whether there is a connection between these places at all; if there is no connection, there is no need to ask the caller for a day or a time.

There are three possibilities for step (4): The informant can give the information required, from a single arrival time to one or more full descriptions of an item (e.g. hotel name and address, single/double room, dates, etc.); the informant can say that there is no information concerning the item specified (e.g. There is no hotel room in the three-star category in Rome during Easter); or, in the case that there is a booking option, the informant can confirm to the callers that s/he now has all the necessary parameters to allow a booking to be made.

For step (5), the informant may have to get additional parameters from the caller, such as number of seats wanted, smoker/nonsmoker etc., and the name of the caller or of the person(s) the booking is for.

Step (6) is dependent on whether a booking is possible and has been asked for by the caller. It should normally coincide with the informant confirming to the caller that the booking has been made.

After both completion and failure of one sequence of these logical steps, both informant and caller can initiate a new start. Some parameters for the completed task can be carried over to the new one and do not necessarily have to be given again.

The booking or reservation facility is an addition to the basic information dialogue. An information dialogue is possible without booking taking place, but not *vice versa*. Thus, all of the applications require a dialogue that can be seen as an instantiation of a common type, namely the Information Providing Dialogue.

[1]The problem is known in expert system research under the terms 'fail-soft behaviour' and 'graceful degradation'

## 2.4 Information providing dialogues

In the previous section, we described the task structure of the SUNDIAL application dialogues in terms of logical steps. These steps are the same for all of the applications, with the exception of the optional steps. It is only the way these steps are realized that differs, that is, what information is available, which parameters have to be elicited in which sequence and so on. This application-specific knowledge can be isolated and confined to distinct modules.

The course of a natural dialogue is not only determined by the task structure. Rather the task structure is only a logical one, and the actual carrying out of the task may require several intermediate steps in the dialogue. There can be repair steps in case of a misunderstanding from one of the parties, confirmation steps to avoid failure, steps for adding to the precision of a parameter and the like, phenomena which are in principle common to all dialogues. It is the combination of both the orientation towards an information task and the special way in which these things are treated that identifies the dialogue type we call information providing dialogue.

Why is it that failure, confirmation etc. are treated differently by people if they want information than in other social or situational settings? Let us have a look at why people call an information service. The caller wants information s/he is unable or unwilling to get for him/herself, (a) because the information is not available where s/he is (like an airline timetable or the knowledge about whether a hotel bed is available), (b) because of difficulties in looking things up (like a train timetable), or (c) because it is easier and/or faster to ask someone who normally has access to it for this information.

In all of these cases, the caller is interested in getting something from the informant. The caller knows that to obtain what s/he wants, s/he has to be co-operative, i.e. make him/herself as clear as possible and give the informant whatever information (or task parameter) the informant says s/he needs to successfully get a result. The caller also knows that the task parameters have to be clear, because if they are not, the caller is likely to be inconvenienced, by for example missing a train or a plane or having no hotel bed at night. Thus, in information providing dialogues, we have the situation that people are willing to go through many steps that would not be made in other settings or at least would cause annoyance.

The informant is also interested in getting his/her job done well. S/he knows that wrong or incomplete information may have some unpleasant consequences for the caller and eventually for him/herself. Thus, s/he will insist on everything being clear and precise and confirmed.

In information providing dialogues, special care is being taken by all of the participants to avoid or repair misunderstandings, to make parameters for the information inquiry as precise as necessary and to be co-operative. For these reasons, this type of dialogue is ideal for a computer speech and dialogue understanding system, where it may be necessary to have confirmation on almost everything because of the difficulties in word recognition.

# Chapter 3

# Dialogue Phenomena

## 3.1 Introduction

### 3.1.1 Aim of this chapter

In this chapter we describe and classify some of the dialogue phenomena which have been observed in information providing dialogues. These dialogues came from different corpora: some are human–human, and others human–machine. The latter were obtained through simulations, in which callers believed they were speaking to a computer speech understanding system, although in fact they were conversing with a person, or a computer under direct experimenter control.[1]

The aim of this chapter is to *describe* what people do in dialogue with each other rather than to *prescribe* what exactly the SUNDIAL system ought to be able to do. Later chapters in this document indicate which of the phenomena described here will be supported by the intermediate demonstrator SUNDIAL system and which, because of limitations in current knowledge and techniques will not be supported. The phenomena are described in a way which is as language independent and task independent as possible. For a few phenomena this is difficult or impossible, and in these cases we present all the main alternative perspectives.

The main conclusion to be drawn from the chapter is the complexity of the dialogue management task and the need for careful description of actual human dialogue.

[1]see [**?**] for more information about these simulations

### 3.1.2 The data

The examples in this chapter are drawn from three task domains and four dialogue corpora:

- Flight enquiries. British Airways (BA) corpus (in English).
- Flight reservation and inquiries. Air France (AF) corpus (in French). The deliverable, [?], describes precisely how this corpus was obtained.
- Flight reservation. Sundial French simulation (FS) corpus (in French).
- Train inquiries. FACIT-Corpus (dberl) (in German).

Examples are labelled by their source corpus and the call number; for example, `BA [7]` signifies that the example comes from the BA corpus, call number seven.

The relationship between the terminology used in this chapter and comparable terminology used elsewhere in this deliverable is described in chapter 1. Also, the dialogue examples we provided here are transcribed according the norm decided in [?], chapter 5.

### 3.1.3 Structure of the chapter

It has been remarked that dialogues are organized in *dialogue phases* [?]. This observation has been intensively used in the French simulation protocol, and also observed in BA, AF and DBERL. This is why we introduce dialogue phenomena within this overall organization in section 2. Within dialogue phases there is a local organization at the level of the exchange or at the level of the utterance. Section 3 deals with such local organization. In parallel to these dialogue structurations, prosodic information plays a role. Section 4 discusses prosodic phenomena in dialogue. Section 5 introduces phenomena tied to informational structures.

*Reading note: paragraphs which begin with a star (*) are of lesser importance. The reader who is not interested in detailed descriptions can skip these paragraphs. These paragraphs mostly describe Human–Human phenomena which do not commonly appear in Human–Machine interaction.*

## 3.2 The Overall Organization of Dialogues

As noted in Chapter 2, each application is task-oriented. This observation motivates the description of information providing dialogues in terms of the succession of the following phases: *dialogue opening, problem formulation, problem resolution,* and *dialogue closing.* Dialogue opening and closing are task independent as we will see. However the two intermediate phases are task dependent.[2]

For information seeking tasks, the ordering of phases often follows the schema [where O represents opening, F is formulation, R is resolution and C is closing]:

O–F–R–F–R–. . . –F–R–C

For the reservation tasks, the pattern can be:

O–F–R–R–. . . –R–C

Now, let us describe each phase as it appears in dialogues.

### 3.2.1 Opening a dialogue

In this section we will consider some of the characteristic features of exchanges which occur at the beginnings of dialogues. As the callers' initial utterances are invariably extended into the request, these will be dealt with as a single 'chunk' of activity.

There are some differences according the type of interaction: human–human vs. human–machine.

**Human–human interaction:** before moving to a formulation of a request for the information they require, speakers regularly produce a variety of items. Some of these are politeness tokens, such as 'good morning' 'hello' and 'I wonder if you could help me'. Other items seem to be used to delay moving to a formulation of the request; for example the 'umm' in BA [1].

[2]For the sake of simplicity we will consider reservation formulation–resolution as two phases even if in fact there are four phases: two for the one-way and two for the return.

| BA[1] | | |
|---|---|---|
| 1 | A: | good morning british airways flight information |
| 2 | C: | *umm I wonder if you can help me please* |
| 3 | | could you check flight bee ay nine oh three... |
| AF[1] | | |
| 1 | A: | air france bonjour |
| 2 | C: | *bonjour* je voudrais faire une réservation pour e |
| 3 | | londres |
| [dberl12] | | |
| 1 | A: | Reiseauskunft Nürnberg Schön Grüß Gott |
| 2 | C: | *Ja, Grüß Gott*, ich bräuchte eine Verbindung füer Sonntag Abend von Hannover nach Nürnberg bitte und zwar so spät wie möglich |

**Human–machine interaction** The opening phase is optional when interacting with a machine.

| FS[2] | | |
|---|---|---|
| 1 | S: | Réservation automatique de billets d'avion<br>Ne raccrochez pas avant le message d'au revoir de Sundial.<br>Formulez votre demande. |
| 2 | C: | *e* je souhaite réserver un billet d'avion… |
| 1 | S: | *system first utterance* |
| 2 | C: | *allo:/,* alors je désire réserver:/ … |

Note that the system directly enters the problem formulation phase with the fragment "formulez votre demande" (*formulate your request*). This may explain the frequent absence of greetings or 'politeness' from users.

Nevertheless, users sometimes enter dialogues with a *greeting*, they may possibly *check the line* (e.g. *allo*), and finally as in human conversation they could use *prefatory* or politeness such as "I wonder if you can help me".

## 3.2.2 Problem formulation

### 3.2.2.1 Service requests

With these items the callers declare what type of service they require from the agent, or indicate their business in ringing the service. For example, in BA[1] the caller says "could you check" (line 3), whereas in AF[1] she says "je voudrais faire une réservation pour". It should be noticed that this is a constant: despite the system saying "formulez votre demande" and "réservation automatique…" in the FS, users introduce their request as shown in the following examples.

| | |
|---|---|
| BA[5]: | 'I'm just phoning to inquire...' |
| BA[18]: | 'I just wanted to check...' |
| AF[31]: | 'je voudrais réserver une place...' |
| AF[4]: | 'j'aurai voulu connaître le prix du billet...' |
| dberl[18]: | 'Könnten Sie mir da einen Zug raussuchen?' |
| dberl[??]: | 'Ich hätte eine Frage' |
| FS[47]: | 'j'aurai voulu:, un aller h, paris lon:dres/...' |
| FS[94]: | 'pour mylène ponamalé:: il nous faudrait un londres paris...' |

**Substantive requirements in information–seeking dialogues:** in addition to formulating the type of service they require, callers may also describe explicitly the information they require. For example, in BA[1] the caller ends the request with 'and I just want to confirm....when it is due in'. Other examples are:

| | |
|---|---|
| BA 3: | ...could you tell me when it will arrive |
| BA 6: | ...could you tell me if you have a confirmed arrival |
| dberl2: | Wann mußt ich da in Nürnberg wegfahren. |

These items may be used in conjunction with service request formulations, as in BA[1], ('could you check..... and I just want to confirm'), or by themselves, as in BA [11].

| | |
|---|---|
| BA [11] | |
| A: | flight information can I help you |
| C: | yes could you tell me the arrival ti- |
| | the expected arrival time of british airways two five eight from caracas please |

### 3.2.2.2 Request details

In addition to nature of the request, callers provide details about the object they have in mind (a flight, a train). A brief overview of the kind of details required for flight inquiries, train inquiries and flight reservation follows.

In the case of BA, AF flight enquiries and FS reservation, the information which can be provided by the caller is as follows (those which concern only flight reservation are in italics):

- Flight Number (bee ay XXXX, A F XXXX,...)
- Single Location Identification. ('...from caracas'; '...from california')
- Dual Location Identification. ( 'flight from geneva to gatwick')
- Airport Details. These often occur as part of Dual Place Identification. For example: 'I... just want to check flights from lyons coming to terminal one'.

- When.
    1. indexicals: ‘tomorrow’, ‘today’, ‘next Wednesday’
    2. parts of days: ‘this morning’, ‘mid day’, ‘this afternoon’
    3. clock or ‘official’ time: ‘eleven thirty’, ‘fifteen fifteen’
- *Class:* economy, business, or first class.
- *Type of travel*: one–way, return, two–ways.
- *Number of tickets* (e.g. number of seats)

Some important observations on the relative ordering and frequency of these items can be made. In flight reservations, the items are the same but of course, ordering and the minimal amount of requested details are different.

The features of the request details in the BA dialogues are different from those in the dialogues on train schedule information. In the FACID dialogues the caller wants information about a specific train connection, and therefore the agent needs information about the goal and the intended departure time.

*** Examples** Not all items are necessarily provided by the callers. In the following extract the caller produces a minimal amount of information, using only the BA flight number and a Single Location Identification:

| BA [8] | |
|---|---|
| A: | yes hello th -ere |
| C: | -oh good (.3) er the bee ay five eight four from turin love |
| A: | five eight four hold on please |

Initial examination of the BA corpus suggests that if the callers know the flight number they will use this before naming the relevant location(s). Apart from this observation it is difficult to predict a neat ordering which the callers use when formulating their requests. Requests of the following kind are not uncommon:

| BA [2] | (line 3 onwards) | |
|---|---|---|
| C: | oh good morning hh | politeness |
| | I wondered | |
| | I’m just checking on a plane | service |
| | that’s due in tomorrow... | day |
| | from warsaw... | location |
| | at half past nine | clock time |
| | on terminal two | airport details |

The above extract illustrates a further issue: if the caller does not know the flight number then the information request is jumbled:

| | | |
|---|---|---|
| BA [47] | (line 3 onwards) | |
| C: | oh hello yes | prefatory |
| | can you tell me please | politeness |
| | if the | |
| | there was a flight | |
| | from new york arriving heathrow | dual location identification |
| | nine o'clock | clock time |
| | this morning | part of day |
| | can you tell me | substantive requirement |
| | if it was there on time | |

In an analysis of train timetable enquiries [?] observed a significant tendency to order request components as follows: place of departure > place of arrival > day > approximate time of travel.

Of the fifty calls in the BA corpus, 32 are requests for arrival times; 28 calls for planes that day, 4 for planes other than that day. There were 13 inquiries about departures; 10 for planes due to depart that day, and 3 for planes due to depart other than that day.

In the FS corpus the bias of the scenarios suggested an ordering of the parameters. However, the observed order is: Number of seats > departure place > arrival place > departure date > departure time > flight number. Departure time is optional depending on whether the caller knows the flight number. However it is not uncommon for the caller to provide both the flight number and departure time.

### 3.2.3 Request resolution(s)

So far little work has been done on the way in which human agents deliver the information requested by the caller. However, some preliminary observations are useful.

It is helpful to consider 'simple' and 'complex' request resolutions separately. A complex request is one which involves the agent having to ask for further information in order to satisfy the request.

#### 3.2.3.1 Simple resolutions

| | |
|---|---|
| BA [50] | |
| C: | could you give me an arrival time please for bee ay three zero nine er which is due from paris at two o'clock |
| | (.5) |
| A: | three oh nine I'll just check for you sir |
| C: | thank you |
| | (8.5) |
| A: | yes the bee ay three oh nine's confirmed at nineteen fifty |
| AF[I6] | |
| A: | je voulais connaitre le tarif étudiant paris ankara |
| C: | paris ankara tarif étudiant c'est bien ça |
| A: | ? s'il y en a: oui ? |
| A: | ne quittez pas monsieur,,, tarif étudiant en aller simple aller retour |
| C: | aller retour |
| A: | aller retour 3240 francs |
| C: | 3240 francs/ |
| A: | c'est bien ça |
| FS[113] | |
| ... | |
| S: | La recherche de vol est en cours; veuiller patienter (18) Oui, paris à 10 heures 35, arrivée à naples a 11 heures 55; ce vol vous convient-il? |
| | (1) |
| C: | oui/, ca m'convient\ |
| | (12) |
| S: | voulez-vous une réservation en classe économique, affaire ou première ? |
| C: | en CLAsse [ekonomig] |
| S: | combien de places voulez-vous réserver? |
| C: | une place\ |

There is a discernible pattern in the way that agents formulate their answers. Invariably the time of the arrival or departure is given first. This may be followed by further items if requested (or, it seems, if the agent simply feels like giving them out.). If these further items are produced they tend to come in a certain order: time, terminal, place. For example:

| | |
|---|---|
| BA [25] | |
| A: | ...that'll be landing at eleven thirty |
| C: | eleven thirty |
| A: | that's right sir terminal four heathrow |

'Simple' calls tend to exhibit the following pattern: a request, the answer and then a swift move to a closing sequence in the case of inquiries. For reservation we have a pair organization: {minimal set of requests, minimal set of answers}.

#### 3.2.3.2 Complex resolutions

A large number of calls do not follow this neat pattern, and are in no way simple. We may distinguish two cases: first, when the caller asks for a variety of different items of information and, second, when the task resolution fails. The following paragraphs describe these two cases.

**Caller motivated** Callers frequently ask for further information on the proposed flight(s) in the reservation task.[3] This kind of complexity is analogous to the complex resolutions in inquiries. Thus we can draw similar conclusions.

| | |
|---|---|
| BA [38] | |
| C: | ... I'd just like to confirm a flight tonight |
| | ahm (.) supposed to be |
| | flying to cyprus on bee ay h six six eight at ten o'clock |
| | (1) |
| A: | right I'll check that fo@r you@ |
| C: | @thank you@ |
| | (4.3) |
| A: | to larnaca |
| C: | that's right yes |
| A: | yes: that's: er scheduled for take off at twenty two hundred |
| C: | right ¿and¡ uh- what time am I supposed to be at the airport to check in |
| A: | yes about two hours before de@parture@ |
| C: | @two hours@ |
| A: | y@es@ |
| C: | @a:nd@ (.) which terminal is that |
| A: | it's the north terminal at gatwick (.) |
| C: | great >dz- dz-< does the railyway |
| | . . . |

| | |
|---|---|
| [dberl9] | |
| A: | Ja, Grüß Gott, ich bräuchte eine Verbindung nach Ulm von Erlangen aus |
| | und zwar gegen 17 Uhr etwa |
| C: | Heute? |
| A: | Ja, heute bitte |
| C: | Ja 17 Uhr 25 ab Erlangen |
| A: | Ja |
| C: | Donauwörth an 19 Uhr 16 |
| A: | Ja |
| C: | Donauwörth weiter 19 Uhr 22 |
| A: | Ja |
| C: | 20 Uhr 18 in Ulm |
| A: | 20 Uhr 18 in Ulm |
| C: | Ja |

[3]Confirmation sequences are dealt with elsewhere (see section **??**).

| | |
|---|---|
| FS[98] | *Confirmation sequences have been deleted from the following excerpt.* |
| S: | . . . oui, paris à 8 heures 55, arrivée à rome à 10 heures 50; ce vol vous convient-il? |
| C: | pouvez-vous répéter l'heure de départ:, s'il vous plait/ |
| S: | l'heure de départ est 8 heures 55 |
| C: | et sur quelle compagnie::, dois-je:: partir/ |
| S: | la compagnie aérienne est air france |
| C: | et: le PRIX, de mon billet, s'il vous plait/ |
| S: | le prix du billet en classe économique est 2475 francs |

These examples are useful as an illustration of the way that callers can continue asking questions on apparently different (but related) topics. They also reveal some of the types of issue other than arrival time and place which human agents may be expected to provide. It remains to be seen what users will expect of a computer agent.

**Task motivated:** a task failure occurs whenever callers' desiderata cannot be fitted. Various level of failure could be distinguished, but this is a matter of cooperation between the agent (system) and the user. When the system suggests another flight, it may be refused or accepted. In the second case, there is a higher probability that the user will request more details on the flight. In the first case they re-enter the resolution problem phase as in the following dialogue:

| | |
|---|---|
| FS[108] | |
| S: | . . . il n'y a pas de vol le 3 janvier, . . . , je vous propose un vol qui part le 5 janvier de paris à 8 heures arrivée à londres à 8 heures 15. Ce vol vous convient-il? |
| U: | non le vol ne me convient pas/\ pourrais, est-ce: >qu<'il est possible de < PARtir/ l:e > DEUX, janvier\ |
| S: | la recherche de vol est en cours; veuillez patienter. . . |

### 3.2.4 Closings

#### 3.2.4.1 In Human–Human communications

Compared to the sequences in which information is given by the agent, the closing sequences are relatively straightforward. In this section we will not try to describe in detail the conditions under which a closing may be initiated and completed; we assume these conditions will vary between task domains, and within domains, depending on the precise nature of the information required. Instead, we will try to illustrate some recurrent patterns which emerge from the BA corpus. In particular, we wish to emphasize that any conversational closing is related not only to the satisfaction of certain conditions, but is also the product of the interactional procedures used by the participants to negotiate a coordinated exit from the talk.

At this early stage we will only point to episodes in the interactions which may be worthy of detailed study. As a guide we will use terminology from Schegloff and Sacks' paper on closings [?]. That is, we will examine closing sequences as a combination of two distinct units: pre-closings and closing exchanges. These aspects are illustrated in the following example.

| BA [11] | |
|---|---|
| A: | terminal four h heathrow airport |
| C: | thank you very much indeed |
| A: | thank yo@u bye bye@ |
| C: | @thank you@ bye |
| [dberl7] | |
| A: | 13 Uhr 56 |
| C: | Ja |
| A: | Vielen Dank |
| C: | Bitte |
| A: | Wiederhören |

Thus we may identify the exchange of 'thank you's' as the pre-closing sequence, and the exchange of 'bye's' as leading to the termination of the call.

We can see some examples of the callers' use of a range of lexical items to begin the pre-closing sequence; for example, 'right', 'okay', 'lovely' and 'great' (BA[10] and [14]) or 'gut', 'wunderbach' (dberl[12] and [15]). These items appear to display the caller's satisfaction with the information provided by the agent.

The significance of the pre-closing sequence is that it allows both participants to establish that there is no other business to be addressed prior to the close of the conversation. There are two sorts of evidence for this. First, it is commonly found that the participants' exchange of 'bye's' overlap, as in BA [10] and [11]. This delicate coordination of 'bye's' indicates that both participants have monitored the pre-closing sequence, inferred that there is no more business to be done, and moved to the terminal exchange at the same moment. Secondly, in the following extract, two pre-closing sequences are initiated by the caller in lines 34 and 36. In both cases, however, the agent continues to give information. After this has been done, the other party initiates yet another pre-closing sequence in line 39. This suggests that the caller is orienting to the completion of a pre-closing as a necessary condition for the termination of the call.

| | | |
|---|---|---|
| BA [8] | | (The caller has just asked the agent how long it will take passengers to go through customs.) |
| 32 | A: | you know you know as well |
| 33 | | as I do yeh i-t could (gu) allow half |
| 34 | C: | -okay love |
| 35 | A: | an hour at any rate |
| 36 | C: | okay dear |
| 37 | | (.) |
| 38 | A: | it may be quicker (.) it may not |
| 39 | C: | okay my love |
| 40 | | (.7) |
| 41 | A: | thank you |
| 42 | C: | thank you |
| 43 | A: | bye bye |

#### 3.2.4.2 In Human–Machine interaction

Closings can be somewhat different in Human–Machine interaction, as a result of the design of the system dialogue, which can be much less subtle than typically found in human dialogue. For example, the system may abruptly terminate the dialogue, failing to offer the caller a pre-closing sequence (as in FS[109] below), or it may even crash, terminating the dialogue in mid-stream.

In FS[107], we see an example of a caller's successful attempt to continue a dialogue even though the system has failed to provide a pre-closing sequence. The hesitations in the caller's utterance signify the difficulties the caller faces in continuing despite the system's untoward closing.

| | |
|---|---|
| FS[109] | |
| S: | Votre demande a été satisfaite, au revoir et à bientôt |
| C: | au r'voir |
| FS[107] | |
| S: | Votre demande a été satisfaite, @ au revoir et à bientôt@ |
| C: | @pouvez-vous me@ |
| C: | s'il vous plait, pouvez-vous me CONfirmer la classe économique/ |
| S: | la classe réservée est la classe économique |
| C: | et:,,, puis-je avoir le PRIX du billet également |
| | ... |

## 3.3 Local Structure

In addition to the global organization of dialogues described in the previous section, dialogues have local organization, from utterance to utterance.

### 3.3.1 Adjacency pairs

#### 3.3.1.1 Description of the paradigm and illustrations

In all phases of the dialogues, participants orient to expectations derived from adjacency pairs. We classify the adjacency pair as a feature of local dialogue phenomena. This is because whatever type of situated and local activity participants are doing - openings, requests, clarifications and closings, and so on - they will orient to the expectations which can be described by the concept of the adjacency pair. Although adjacency pairs were initially introduced by [**?**], we will refer here to Heritage's discussion ([**?**], pp.245-253). Many conversational exchanges occur as paired actions: greeting-greeting, request-acceptance/ refusal, and so on. Sacks developed a generic analytic apparatus, the adjacency pair, to handle these actions. According to the [**?**] formulation, an adjacency pair is a sequence of utterances which are:

1. adjacent,
2. produced by different speakers,
3. ordered as a first part and second part,
4. typed, so that a first part requires a particular second part (or range of second parts) (Heritage 1984: 246.).

Some important points need to be clearly stated. First, the concept of the adjacency pair does not claim that paired actions are necessarily produced as succeeding actions which occur next to each other. Neither is the concept used to try to capture some empirical generalisation, for example, that in 80% of cases second parts immediately follow first parts. Rather, the concept is primarily useful in the way it highlights the normative character of paired actions. That is, a speaker's production of the first part of a pair proposes that a relevant second part is expectable. Moreover, because of the expectations generated through the use of the first part, the participants have the basis for inferential work about each others' actions. For example, if A says 'hello' and B does not reply, A might think that B was deliberately rude, or it may be that B had not heard. A would not conclude simply that a statistically unlikely event had just occurred.

Some of these issues can be illustrated in the following extract.

BA [7]

$$\left\{\begin{array}{ll} C: & \text{are you able to tell me (.5) er flight arrival times from zimbabwe oh no you} \\ & \text{can't i- its at gatwick (.5)} \\ A: & \textit{no} \end{array}\right.$$

$$\left\{\begin{array}{ll} & what's\ the\ flight\ number\ (1) \\ C: & urm\ (.) \end{array}\right.$$

$$\left\{\begin{array}{lll} A: & whAT's\ thE\ flIGHT\ nUM & -ber \\ C: & & -flight\ number \\ & I\ don't\ know\ the & flight\ number\ it \\ & arrives\ saturday & morning \end{array}\right.$$

The first part of the second pair sets up the expectation that an answer should follow. Clearly, the caller has some difficulty with the question insofar as there is a one second pause (although the precise nature of that difficulty is not clear). What the agent does next is to repeat the question with stress on certain parts of the words. The agent's repeat is informed by the expectation that an answer should follow a question. We may speculate also that the added emphasis on certain words also suggests the agent's reasoning as to the source of the caller's difficulty: she is treating the problem as one of 'not having heard' rather than 'not understanding the question'.

The caller overlaps with the end of the agent's repeat (third pair). This is a 'collaborative completion', that is, where the caller's overlap correctly projects the word(s) being used by the agent. This displays that the caller recognises and anticipates the 'gist' of the agent's repeat. Furthermore, by 'echoing' an essential part of the original question, the caller displays that he had heard it, and understood it. His subsequent utterance 'I don't know the flight number' provides an account for why he had not answered the question; this account - his lack of knowledge - is clearly different to the one proposed by the agent's repeat - that he had not heard the original question.

In this example, then, it seems that the speakers are orienting to the types of expectations which are described by the adjacency pair concept. The agent's repeat of the question is informed by the absence of an answer to her question, and her repeat displays her reasoning as to why there was no answer. The caller's subsequent utterance displays that the agent's understanding was wrong, and provides an alternative account for why no answer had been provided. This indicates his orientation to the normative expectations generated through the provision of first pair parts.

The normative character of adjacency pairs can also be demonstrated by looking at occasions in which the expectations proposed by a first pair part are held over sometimes lengthy stretches of inserted talk. For example, in the following extract, taken from the beginning of a telephone call, there is a 'nested' sequence of question-answer pairs within a question-answer pair. Note that the resolution of the exchange results in a second pair part which is tied to the first pair part which is B's first utterance.

| | | |
|---|---|---|
| A: | Hello | |
| B: | Is Fred there ? | Q1 |
| A: | Who is calling ? | Q2 |
| B: | Is he there ? | Q3 |
| A: | Yes | A3 |
| B: | This is J. Henderson | A2 |
| A: | Just a moment (goes to fetch) | A1 |

In the following example the insertion sequence is slightly different in that there is no 'neat' nesting of other paired parts.

| BA [10] | | |
|---|---|---|
| 6 | C: | I'd like to check on the |
| 7 | | arrival time of bee ay zero eight four |
| 8 | | (.) uhm: vancouver seattle (.) to heathrow |
| 9 | | (.) |
| 10 | A: | today sir |
| 11 | | (.) |
| 12 | C: | today |
| 13 | A: | thank you sir hold the line |
| 14 | C: | thank you |
| 15 | | (.) |
| 16 | A: | yes the flight's on route it's expected now |
| 17 | | at fifteen fifteen |

In this example the caller's initial inquiry occurs in lines 6 to 8. The agent's next utterance is an inserted inquiry seeking clarification of the details for the relevant flight. After this insertion sequence the agent answers the original query by providing the required information.

The following example further indicates that participants orient to the normative character of adjacency pairs, rather than the statistical probability of the relationship between paired parts.

BA [13]

| | | |
|---|---|---|
| 1 | A: | flight information british airways good day |
| 2 | | can I help you |
| 3 | | (.) |
| 4 | C: | h erm yes I wonder if you could tell me |
| 5 | | I haven't actually got the flight number |
| 6 | | but there should be a flight coming in |
| 7 | | from crete this morning (.3) and I'm |
| 8 | | wondering what time it arrives |
| 9 | A: | right where at gatwick |
| 10 | | (.) |
| 11 | C: | er yeah (.) -sorry |
| 12 | A: | -any idea of a time |
| 13 | C: | hh I think it's roughly ten o'clock |
| 14 | A: | right hold on |
| 15 | | (43) |
| 16 | A: | you're probably find it's an air |
| 17 | | tours flight |
| 18 | | (1) |
| 19 | C: | right |
| 20 | A: | a charter flight erm: (.3) but I'm just |
| 21 | | trying to find out for y-ou |
| 22 | C: | -huhh hah |
| 23 | | hh |
| 24 | A: | it's definitely into gatwick at ten |
| 25 | | it may not be a flight we handle |
| 26 | | at all just a moment |
| 27 | C: | okay |
| 28 | | (73) |
| 29 | A: | well I'm sorry I've looked up (.) |
| 30 | | heraculen which is crete |
| 31 | C: | yeah |
| 32 | | (.) |
| 33 | A: | and chania we don't seem to have |
| 34 | | any thing you you may well find |
| 35 | | it's another carrier |

In this extract the caller has asked for information about a flight which, it transpires, is not handled by BA. The first point to note is that there is a lot of clarification work in lines 12 to 28. Yet the expectation that an answer is forthcoming is held over, and indeed, in line 29 the agent does address this expectation by giving an account for why she can not produce an answer. A second point is that the agent's account is formulated as a typical dispreferred answer. That is, it is marked by items such 'well', an apology and an account for why she cannot provide an answer. Therefore this extract illustrates two sources of evidence for the normative expectation that an information request should be returned by the relevant details: the fact that the agent does address the caller's request, despite the considerable insertion sequence; and that the failure to provide a 'proper' answer is given in a dispreferred format.

As Levinson proposed, second parts may be classified into *preferred* and *dispreferred* categories [**?**]:

| *FIRST PARTS* | | | | | |
|---|---|---|---|---|---|
| | Request | Offer/Invite | Assessment | Question | Blame |
| *SECOND PARTS* | | | | | |
| *Preferred* | acceptance | acceptance | agreement | expected answer | denial |
| *Dispreferred* | refusal | refusal | disagreement | unexpected answer or non–answer | admission |

#### 3.3.1.2 Arguments against the adjacency pair paradigm

Despite the elegance of such paradigm, it can be argued that many dialogue phenomena cannot be analyzed in term of pairs. First, some dialogue interventions cannot be integrated in any pair and require a triplet organization, and second it may happen that a first part of a pair will never be completed with its second part and thus the pair paradigm is invalidated. The two following examples illustrate these arguments:

| FS[100] | |
|---|---|
| C: | A, QUELLE DATE de depart: e |
| S: | le 4 janvier |
| C: | *D'accord*\ |
| S: | Désirez-vous un retour |
| | . . . |

| FS[98] | |
|---|---|
| S: | *. . . ce vol vous convient-il?* |
| C: | pouvez-vous répéter l'heure de départ:, s'il vous plait/ |
| S: | l'heure de départ est 8 heures 55 |
| C: | et l'heure d'arrivée/ 10 heures 50\ |
| S: | oui |
| C: | #h pouvez-vous me confirmer la:: DAte, du départ/ est-ce bien, le NEUF février\ |
| S: | oui |
| C: | >et< sur quelle compagnie::, dois-je:: partir/ |
| S: | la compagnie aérienne et air france |
| C: | pardon:/ |
| S: | la compagnie aérienne et air france |
| C: | et: le PRIX, de mon billet, s'il vous plait/ |
| C: | #h, voulez-vous, que je vous laisse le:: nom: de::, la personne, qui réserve/ le billet\ |
| S: | voulez-vous une réservation en classe économique, affaire, ou première |
| | . . . |

"D'accord" means that the caller is satisfied. Thus this is a kind of triplet

<Question ; Answer ß Statement ; Agreement >. This is what motivated Roulet's theory of dialogue organization [?] and why he introduced the triplet possibility qualifying the first argument as "initiative" (the move which creates an exchange), the second argument as "reaction" and the third optional argument (and thus this includes the adjacency pair model) as "evaluation". In this model, reactions may be postponed, giving an embedded exchange situation (analogous to the embedding possibilities of adjacency pairs) and evaluations could be evaluative exchanges (which relates the so-called: co-oriented argumentation) which are conditions for closing the exchange definitively (in saying "d'accord" the caller explicitly agrees about this parameter. It is then doubtful whether questioning could occur on this parameter in the rest of the dialogue).

In FS[98] the problem is that "ce vol vous convient-il" will never be *explicitly* answered. This phenomenon appears also in human conversation. In our example many incident exchanges occur; these subexchanges are related to the flight in consideration and as far as the caller doesn't express any surprise or disagreement the expected answer is implicitly given and interpreted as "yes".[4]

The controversy described above is not intended to make us adopt any particular linguistic model (adjacency pairs vs. Roulet's model). In the spirit of exposing dialogue phenomena it is interesting to discover whether there is a framework or not in which observations neutral with respect to either approach could be made. As we stated implicitly, Roulet's model is not enough (see discussion on example FS[98]) and probably none of the existing models is generic enough. For the sake of respecting theory neutralness, we will use the word "sequence" instead of any engaged terminology such as "pair" or "exchange".

### 3.3.2 Turn taking

Independently of the local organization, the very first observable feature is that each dialoguee is allowed to speak. Waiting for the end of the other dialoguee's utterance to take a turn is a feature of politeness. But turn taking rules are not rigid: even if the two locutors are polite they may speak in overlap or interrupt the other, especially in human–human conversation (see section **??**).

#### 3.3.2.1 Turn giving

**Intonation:** the fact that a turn is being offered to the hearer can be signalled by the use of a number of different turn completion or response ellicitation markers. One of these markers is **intonation**. Much work remains to be done in the

[4]Conversely, if there is a disagreement then this is an implicit negative answer.

analysis of intonation but some clear examples of its turn giving function can be seen in the openings of calls in the BA corpus. Typically the agent's first turn includes a service identification such as "british airways flight information". Optionally this can be followed by an explicit offer such as "can I help you". There is hardly ever a perceptible pause between the end of the agent's turn and the beginning of the caller's turn, regardless of whether the optional element is present or not. This is because the offer of the turn is signalled by a fall-rise intonation pattern on the last two syllables of the speaker's turn rather than by the lexical content of the turn. However, this pattern seems to be a feature specific to English.

In the Human–Machine interaction simulations, the system had no intonation, or at least the intonation was not clear to the caller. Nevertheless, callers still used intonation as though they were speaking with a human. For example, in FS corpus, there is a rise-fall pattern in the caller's first utterance, when they are taking the first step in problem formulation.

**Question tags:** it is often assumed that **question tags** such as, in English, *isn't it, doesn't it, aren't they,* etc are reliable markers of the end of turns. However, this assumption finds no support in the *BA corpus.* Only two question tags have been found in the entire BA corpus and both of these are located in off-line asides to colleagues of the speaker. In neither case does the question tag mark the end of the turn (BA[28]: "unbelievable isn't it it takes so long", and BA[7]: "it is the twenty third (isn't) it margeret (.7) yes"). A much more reliable marker in English of the end of a turn is the kind of **offered answer** seen in the example above where the speaker asks a question and then prompts the hearer to answer by offering a candidate answer "yes". Here are some further examples: "are you a real person yeh(t)" (BA[8]), "yeah you yu- you use them too yeah" (BA[28]).

However, in German, there were six cases in which question tags signalled turn giving. The question tags were: nä[5], gell, and nicht war. Each of these tags imply that the hearer should make some reaction. For example, the question tags could mark requests for confirmation.

In French, the academic question tag "n'est–ce pas" is rare. In the Air France corpus, the tag "hein" is the one most often used. In the FS corpus no tag corresponds exactly to "n'est-ce pas". However, questions were be marked with politeness formula, such as "s'il vous plait" and "merci": these two markers were often used in FS (whereas they are quite rare in the AF).

[5]Nä means nicht war; the corpus is dominated by southern German variants, of which this is an example.

**Turn keeping:** Speakers adopt a number of strategies for keeping a turn when it would otherwise be possible for the hearer to begin a new turn. One such strategy is to **fill hesitations** with non-lexical fillers such as “er” and “um”. In the following example the caller has asked to have a message passed to a passenger on a plane. The agent asks if the passenger knows the caller. The “erm” may signal that the subsequent silence is not to be filled.

| BA [3] | | |
|---|---|---|
| 129 | C: | yes er:m: (.7) his s- well say his son david |

In the BA example below, the caller has a complex query to put across and there are a number of places where he could forfeit his turn. He uses several non-lexical fillers to hold the turn. Notice also the use of the lexical filler “sort of”.

| BA [29] | |
|---|---|
| C: | oh hello (.) erm I’ve been asked to *erm* (.3) warn someone and I’m not quite sure where to start but three passengers on a british airways flight to *er* (.) new york will be cutting it a little bit fine this evening *uhm* but they should be there within *sort of* (1) couple of minutes of *uhr:m* (.7) check in time |
| [dberl 16] | |
| C: | wann geht denn der *äh* davor |
| AF[I88] | |
| C: | *alors e* je voudrais deux places / pour le *e:* 27 |

Another turn keeping strategy involves speeding up the speaking rate at potential turn transition points. The following examples show some **rush-throughs** of this kind.

| BA [21] | |
|---|---|
| | thirteen thirty |
| C: | thirteen thirty>is that confirmed |
| A: | yes it is |

* **Turn grabbing:** In face to face interactions a speaker can signal that he wants to ‘grab’ a turn by breathing in audibly. On some occasions an **inbreath** will force a speaker to relinquish the turn to the hearer. It remains to be seen whether or not inbreaths serve the same function in telephone interactions.

The most effective way to grab a turn is simply to start speaking. The following examples illustrate this:

| BA [13] | | |
|---|---|---|
| 79 | A: | you know I wish I could help you |
| 80 | | a little more b-ut |
| 81 | C: | -nehver mihnd hh alright lovely |

However, there is no guarantee that the speaker will relinquish the turn when interrupted. The speaker has the option of talking through the interruption and holding the turn.

| BA [23] | | |
|---|---|---|
| 19 | A: | it is a b- wh-at I'm trying (n) explain |
| 20 | C: | -it's actually (? ) |
| 21 | A: | it's a bookable service rather than a turn up |

Sometimes both speakers can talk in **overlap** with no indication that one speaker is trying to wrest the turn from the other. For example, simultaneous turns are often found in the closing sections of conversations.

| BA [29] | | |
|---|---|---|
| 66 | C: | yes I'll do that |
| 67 | | thank y-ou very much indeed |
| 68 | A: | -okay you're welcome |

* **Continuations:** There is not a one to one correspondence between utterances and sentences. Thus it is not valid to say that the end of a sentence is a reliable indicator of the end of a turn. In any case the end of a sentence can only be identified after the next sentence has begun. Before that it is always possible to extend the last utterance indefinately by the addition of conjoined phrases or clauses.

Sentences and turns interact in a complex way. In some of the examples given above, sentence endings are seen to correspond to ends of turns. In the following example, the agent is supplying some information and the caller says "yeah". The agent then resumes by finishing her sentence. This is a **same-speaker continuation**. The caller's "yeah" could be termed **turn allowing** in that it is located within a turn without being either an overlap or a turn grabbing interruption.

| BA [13] | |
|---|---|
| A: | the only thing is it could be nother carrier you see and this is not unusual there's many many charter flights now (.) |
| C: | yeah |
| A: | that operate in and out of er (.3) gatwick |
| [dberl 21] | |
| A: | da geht alle Stunde der Intercity |
| C: | mhm |
| A: | immer ab Köln um drei also |
| AF[I92] | *the caller is an agency* |
| C: | le retour fk570 du 9 février |
| A: | hem |
| C: | départ de Charles de Gaulle numéro 1 à 19 heures 50 |
| A: | hem |
| C: | arrivée 21 heures 55 à Oslo |

Sometimes a sentence begun by one speaker is finished by the other. The next example shows such an **other-speaker continuation**.

| BA [3] | | |
|---|---|---|
| 18 | A: | yeah(p) hold on (1) it was due at |
| 19 | | ten er |
| 20 | | (.) |
| 21 | C: | twenty five |

*** A collaborative completion** happens when the hearer joins in to complete the speaker's utterance by anticipating what comes next. In collaborative completions both speakers say the same thing at the same time, although it is not unusual for the hearer's words to come slightly before those of the speaker as in the following example.

| BA [4] | |
|---|---|
| A: | hello the two nine two landed i(-) you did say the |
| | - two nine two didn't you |
| C: | -two nine two yeah |
| [dbt 12] | |
| C: | und die andere Möglichkeit als über Karlsruhe zu fahren gibt es nicht eine günstige |
| A: | ja zu dem Zeitpunkt |
| | –ist es schlecht |
| C: | –ist es schlecht |

**Echoes** A hearer often echoes exactly what a speaker has just said if this is a number or a time. The original speaker tends to confirm whether or not the echo was true to the original.

| BA [4] | |
|---|---|
| A: | yes it landed at six fifty fiv-e |
| C: -six fifty five | |
| dberl[10] | |
| A: | ja in Ulm fünfzehn Uhr siebenundfünfzig |
| C: | fünfzehn siebenundfünfzig |
| AF[I31] | |
| A: | il y a bien 18h ou bien: 20h30 |
| C: | 20h30 |
| A: | ou également 16h25 |
| C: 16h25 ,,, 18h/,, 18h oui | |

However, some echoes are more subtle than this. Rather than echoing the members of a list immediately, the participants store and later re-use verbatim phrases, descriptions or even clauses. For example, the description "ahead of schedule" is used six times in the conversation from which the following example is taken.

| BA [3] | | |
|---|---|---|
| 14 | A: | yes it(ll) be landing now (.3) ahead |
| 15 | | of schedule let me just check that for you |
| 16 | C: | ahead of schedule -bee ay two nine six |
| 17 | A: | -ahead of schedule (.) |

**Sentence mood** The relationship between sentence mood and dialogue act is not clear. While there is certainly some correlation between indicative mood and dialogue acts such as answers, it is not a reliable correlation. Indicative sentences are often used to ask questions (so-called *declarative questions* (*I was wondering..., I'm just checking...*), question forms are used to express imperatives (*would you hold on*), etc. In addition to these problems, many utterances in talk carry no mood at all (*just a moment, okay good enough, what today, air zimbabwe, in two weeks time*, etc). While sentence mood is clearly used in a subtle and sophisticated way to signal a variety of things (including dialogue act, politeness, and temporal reference), a great deal of empirical work still remains to be done before the phenomena are understood.

### 3.3.3 Overlap and Interruptions

It would require a considerable amount of analysis to distinguish clearly between simple overlaps and overlaps which are interruptions; consequently, in this paper we merely provide a basic description of some instances where both participants speak at once. Levinson ([**?**], p296) said that "less than 5 per cent of the speech stream is delivered in overlap". Overlapping talk, however, seems to have some recurrent features. Overlaps can be classified as follows [**?**] (these 'types' may not be mutually exclusive). There may be some overlap between overlap types!

*** Transition Space Onsets** These are cases where the next speaker starts up at a transition relevance place anticipating that the current speaker will not continue. A transition relevance place is located between turn construction units, that is, information chunks.

| BA [7] | |
|---|---|
| A: | certainly can you hold the line please (2) two eight six from |
| | san francisco @is on its@ way expected |
| C: | @that's th@ |
| [debrl 12] | |
| A: | fahren bis Saal mit dem Zug da sind Sie dann neunzehn Uhr sechsundzwanzig |
| | @und von dort achtz@ehn Uhr achtzehn |
| C: | @vorher geht nichts@ |

* **Anticipation of turn completion** and hitching onto the end

| | |
|---|---|
| BA [29] | |
| C: | nine two oh six (.7) should I ask for anyone in particu@lar@ |
| A: | @no:a:an (.) they're going one the one seven nine |
| [dbf 13] | |
| C: | an dem Tag wo ich losfahre also wenn ich noch in eine teure Zone reinkomme |
| | gilt das für mich @nicht mehr@ |
| A: | @entscheidend ist@ |
| | der Tag an dem Sie losfahren |

* **Last part of a turn construction unit** but the speaker continues

| | |
|---|---|
| BA [3] | |
| A: | yes it(ll) be landing,,, ahead of schedule let me just check that for you |
| C: | ahead of schedule @bee ay two nine six@ |
| A: | @ahead of schedule@ (.) |
| | yeah(p) hold on (1) it weas due at ten er |
| [dbt 3] | |
| A: | genügt es wenn Sie um achtzehn Uhr fünfzehn in @1Aachen ankommen@ |
| C: | @1ja das@ @2genügt@ |
| A: | @2ja eine@ günstige Möglichkeit wäre dann ab Regensburg um zwölf Uhr achtundvierzig |

* **Last Item onset** There are two varieties of this. In the first, the overlapping talk indicates that the speaker recognizes the item being said.

| | |
|---|---|
| BA [47] | |
| A: | (hh)we had a bee ay one seven six |
| | which was due in at eight fifty five |
| | this morni-@ng@ |
| C: | @-oh@ that's probably the one |
| [dbf 17] | |
| A: | ja der kürzere Weg ist schon über –@ Bebra Altenbecken @ |
| C: | –@ ja über Bebra ja @ |
| | dann nehme ich den ja über Bebra |

In the second the speaker displays recognition of the thrust or gist of the current speakers turn.

| BA [46] | | |
|---|---|---|
| 6 | A: | can I help you at a@ll sir@ |
| 7 | C: | @oh yes @erm: I've (got a |
| 8 | | note) here to sort of find out about ehm: |

*** Recognitional Onset** These are examples of deeper incursions into the current speakers talk rather than just last item recognitional onsets.

| BA [41] | |
|---|---|
| A: | well that is at the moment yes if you'd like to give us a ring back about |
| | say (.) say about three o'clock we should be able to |
| | confirm it hopefully or let you know |
| | what the l@atest situation is then@ |
| C: | @right but (th-) but earliest@ arrival will be twenty one fifty it |
| | can't be before that |
| [dbt 5] | |
| A: | bis zu zwei Wochen oder drei Wochen im voraus können Sie |
| | die @Fahrkarte schon lösen@ |
| C: | @kann man die schon @holen@ |
| A: | @ja ja@ |
| C: | also die könnte ich heute schon holen wenn ich am Freitag wegfahre |

*** Minimal Continuers** These are cases where the agent interrupts the caller on a minimal continuer such as 'er:', 'hm', 'mm', 'erm', 'hein', 'euh'.

| BA [7] | |
|---|---|
| C: | excellent thank you very much and what you're saying is that I should |
| | phone up er-: |
| A: | -on that number that |
| [dbf 13] | |
| C: | wann ist der günstigste Tag wann kann man da am längsten fahren |
| | da mit rosarot das @äh@ |
| A: | @wie@ der günstigste Tag was |

*** Pauses** There are cases where the current speaker clearly has not finished but pauses, and in the pause the next speaker begins.

| | |
|---|---|
| BA [14] | |
| C: | hello yes I'm just inquiring about the: uhm bee ay flight two two seven (.5) @uhm from@ |
| A: | @two two seven @ |
| [dbf 13] | |
| A: | Hannover kommen Sie an zwölf Uhr vierzig,, ab zwölf Uhr sechsundfünfzig ,, in Osnabrück vierzehn Uhr elf –. . |
| C: | –und wenn ich hier erst Mittag fahre daš es einen Zug durchgibt von Würzburg nach Hannover |

Only two instances of overlap have so far been observed in French simulation (FS) dialogues. These were:

1. when the system says "veuillez patienter, la recherche de vols est en cours", callers often say "merci" in overlap;
2. when the system produces a confirmation in echo (see section on confirmation sequences below, **??**), callers may protest in overlap but then reutter their protestation when they have unambiguously obtained the turn.

### 3.3.4 Failure–repair

To try and give a formal definition of what is a failure implies to have a formal model of what is a "normal dialogue" that is a dialogue without failure. Failures designate all type of hitches that could occur in a dialogue. Repairs refer to "procedures" (e.g. dialogue sequences) triggered to solve the associated failure.

Now we will explore some of the types of failure found in the data and give, whenever possible, the associated repair process. We may distinguish different types of failure–repair couples: *self–failure/repair, language–failure/repair, acoustic–failure/repair* and *domain–failure/repair.*

#### 3.3.4.1 Self–failure/repair

The majority of these involve linguistic knowledge. Two types were observed. In the first type there is some degree of semantic inconsistency or anomaly between the erroneous phrase and the repair phrase.

| | |
|---|---|
| BA [2] | |
| A: | ah well you've actually wro(t)-rung the wrong number |
| [dberl 17] | |
| C: | ich bräuchte für morgen eine Verbindung von äh Nürnberg nach Köln und zwar –nein ich brauche von Nürnberg nach Dortmund |
| AF[II12] | |
| C: | ok on va mettre prendre –on va les mettre en liste |
| FS[114] | |
| C: | non/, le retour n'est pas [prE], retour non, pas prévu, #h par [ko ], en revan:che je veux le PRIX du billet |

Such failure–repair could be characterised by <meaningless part, meaningful part>. Here, 'meaningless' does not necessarily mean that the speech is 'none-sense', but merely that that the speaker is not satisfied (because of perceived linguistic, semantic or intentional inconsistencies).

#### 3.3.4.2 Domain and task–failure/repair

Repair sequences can be classified in terms of who initiates the repair and who carries it out. While there is a general preference for self-initiated self-repairs (where either participant could repair the error), domain or task failures are systematically initiated and repaired by the agent (the human operator or the machine).

However, repair processes are not the same with human and machine.

**Human repair strategy:** the most common repair strategy for the agent is next turn repair initiators such as Wh-questions, negation of alternative possibility and negation of topic with optional modulators such 'I think you mean . . . ' with a plausible alternative

| | |
|---|---|
| BA[8] | |
| A: | er we don't have five eight four sir I think you might mean the five seven nine |
| BA[24] | |
| C: | er::m yes I'm actually trying to make an inquiry about (.) gibralter airways (.) which I think you (.) handle as well (.5) |
| A: | er not on this number sir it's handled on seven five nine one eight one eight |

The repair itself typically takes the form of a repetition of the relevant word or phrase, re-phrasing of a request or clarification.

Task failures concern knowledge on which the agent has no information; such as Gibraltar Airways for BA or AF. In both, the agent tried to give a phone number, that is a cooperative answer. Domain failures mainly concern misconceptions on parameters from the caller. In BA, other task failures occurred (such as a caller asked for confirmation that seats were available on a flight [BA26]) and in each case the agent gave the phone number of the appropriate service.

**Repair in the Air France corpus:** Mary–Annick Morel [?] identified different ways of presenting information by the agent (using a variety of linguistic formulations "c'est ..." *it is ...*, "il y a ..." *there is/are ...*, "nous avons ..." *we have ...*, "vous avez ..." *you have ...*, "j'ai ..." *I have ...* and their corresponding negatives). In case of a failure, the failure denotation is $< presentative >$ and the repair is introduced by the word "mais" (but) or "en revanche", "par contre" followed by the correct clarification.

Conditions of using such formulation are stated according to the level of the nature of the erroneous knowledge, depending on the following conditions:

- if it is a permanent proposition, e.g. tied or not to the company;
- if it is a temporal point;
- if we are in the beginning of the dialogue (that is problem formulation in our terminology) or not;
- if the agent is implicated or not (e.g. human emotion).

**Repair strategy in the FS corpus:** The accomplice in the French simulation was not provided with any means of giving cooperative answers when a task failure occurred. He just triggered the following sentence: "la base de données ne contient pas cette information. Veuillez reprendre le dialogue". This feature was used when users wanted information on trains, or information on the airplane ticket (when they will receive it and so on) which is not part of the envisaged functionality of the Sundial system. A similar response was made when the database was empty about a some destinations.

#### 3.3.4.3 Acoustic–failure/repair

**In human–human interaction:** few acoustic errors have been observed in the BA, AF and FACIT corpora. In all cases, the repair sequence is initiated immediately.
The stress on the expressions in BA [39] is perhaps a consequence of their acoustic similarity.

| BA[38] | |
|---|---|
| C: | great>dz- dz-< does the railway take me to the north terminal |
| A: | I beg your pardo-n |
| C: | -the railway take me to the north terminal |

Although we have not observed them, it is possible for there to be acoustic errors which are not immediately recognized and repaired. For example, the agent may interpret an acoustic sequence as "Toulon" rather than "Lannion" as intended and access information about Toulon flights. Only when this information is conveyed to the caller will the error be detected and a repair sequence initiated.

**In human–machine interaction:** the system may fail to recognize a word or a sentence, or the caller may fail to recognize a word or a sentence.

In the FS protocol, the accomplice was required on occasion to misunderstand words and also to ask the caller to repeat her utterance. Callers were responsible for the repair of misunderstandings. The following excerpt is typical of the interactions found in the corpus:

| FS[139] | |
|---|---|
| S: | paris - mulhouse: quand voulez-vous partir? |
| C: | non\, c'est paris, < TOUlou:se > |
| S: | paris - toutlous: quand voulez-vous partir? |

Several different repair patterns were observed, such as "le dix-neuf janvier et non pas le dix-huit" *january the 19th and not the 18th* (dialogue FS[57]), and more verbose versions such as in "non je crois que vous avez compris seize heures et moi c'est sept heures du matin" *no, I think you've understood 16 hours and I mean 7 in the morning*, (dialogue FS[42]), where "seize" and "sept" are similarly pronounced in French The user provides a disambiguating adjunct "du matin" *in the morning.*

The accomplice forgot once or twice to confirm parameters and this led to interesting exchanges, as for example in the following dialogue:

| | |
|---|---|
| C1: | #h je voudrais-_-un::, PAris ro:me, PAris::, #h avec un depa:rt, de PAris:, le [kat] fevrier::,, a, VINGT-heu::res/ et un depart,/ de ro::me,\ le:: SEPT, fevrier, a dix-neuf-heures trente-cinq/ |
| S2: | La recherche de vols est en cours; veuillez patienter Il n y a pas d'aéroport a bonn, l'aéroport le plus proche se trouve a Cologne, je vous propose un vol qui part de paris a 18 heures 25 arrive1e a Cologne a 19 heures 25. Ce vol vous convient-il ? |
| C2: | n:on/,, je voudrais,-_-un::, paris, RO, ME\ (3) avec un depa:rt, le [kat], fevrier/, a, vingt-heu:res, de, PAris / |
| S3: | La recherche de vols est en cours; veuillez patienter Oui, paris a 20 heures, arrive1e a rome a 22 heures ; ce vol vous convient-il ? |
| U3: | oui:/ |

Having failed to confirm the destination parameter, the system went immediately to the database (S2) and did a complex search using the wrong parameter. One can understand the anguish of the caller in C2 who felt that she had to repeat all the parameters.

The other side of the coin is that the caller may not understand what the system says, especially when the synthesis board pronounces figures (as with flight numbers) or proper nouns (the names of companies, for example). In such cases the failure detection is done by the user who simply says "pouvez–vous répéter" possibly adding the type of the parameter "le nom de la compagnie" or "le numéro de vol". The simulation protocol did not allow for repair in these circumstanes; the accomplice was sometimes able to improvise by spelling the parameter in question (see **??**). Otherwise, when user asked for repetition, the system simply repeated its sentence and this seemed satisfactory.

### 3.3.5 Confirmation sequences

In oral dialogue there are occasions in which the agents seek confirmation that they have heard correctly the details of the caller's request. Confirmation sequences can prevent delayed acoustic failure detection. Confirmation may be accomplished in two ways.

**Confirmation echo:** these occur immediately after the completion of the request formulation. The agent may repeat a variety of relevant details; this repetition is a confirmation which doesn't oblige the hearer to react on it unless there is a misunderstanding.

| | |
|---|---|
| BA [49] | |
| C: | stockholm bee ay seven eight five |
| | (.3) |
| A: | *seven eight five* I'll just check for you sir |
| [dberl20] | |
| C: | ich hätte eine Frage ich möchte am nächsten Sonntag nach Hamm fahren |
| A: | *nach Hamm* |
| AF[I31] | |
| C: | je voudrais une réservation pour Brest |
| A: | *pour brest* |
| FS[141] | |
| C: | bonjou:r/, je voudrais faire, une rśervation:/, pour le, QUAtre janvier::\, #h un, paris lon::dres, depa:rt, SEPT-heures du matin\ |
| S: | *paris londres le 4 janvier* a quelle heure voulez-vous partir |

* **Post-pause confirmation echo:** this confirmation of the caller's details occurs after the agent has checked the flight information database. This type of confirmation was not used in the FS design, thus we have no human–machine examples.

| | |
|---|---|
| BA [18] | |
| C: | I just wanted to check if (.3) flight one seven three (2) from gatwick is still ur:m: (.) mid day |
| A: | right hold on please (1.7) *the one seven three gatwick to new york* |
| C: | yeah |
| [dberl15] | |
| A: | ich hätte eine Frage ich möchte am nächsten Wochenende von Hannover zurück nach Nürnberg fahren. Wann geht denn da der letzte IC ? |
| C: | der letzte ? |
| A: | der letzte ja |
| C: | Moment *bitte von Hannover aus* ? ja ? |
| A: | ja |

### 3.3.6 Clarification sequences

Most of the clarification sequences initiated by the agent attend to details which the caller did not include in the information request. We may term these 'missing parameters'. These clarifications are initiated by the agent or the system directly after the completion of the request. For example:

| | |
|---|---|
| BA [42] | |
| C: | flight (.3) from geneva to gatwick (.5) |
| A: | geneva to gatwick do you have a flight number madam |
| [dberl6] | |
| A: | Ich hätte gerne eine Auskunft und zwar möchte ich heute noch nach Wiesbaden fahren |
| C: | Wann wollen Sie fahren? |
| AF[I31] | |
| C: | je voudrais une réservation pour Brest |
| A: | pour brest |
| C: | oui |
| A: | vous désirez partir quand |

In these clarification sequences the agent requests further information.

### 3.3.7 Co-operative answers

In this section we present phenomena which respect two of Grice's maxims [**?**]:

- Maxim of quantity: make the contribution as informative as desired but not more so.

- Maxim of manner: avoid obscurity, avoid ambiguity, be brief, be orderly.

A co-operative answer is one in which a participant either supplies relevant information additional to what was requested or suggests a re-formulation of a problematic request. It is difficult to know the necessary amount of additional information (maxim of quantity). Another problem is to know when and how (maxim of manner) to provide the further information. Nevertheless, an *a posteriori* analysis of non-cooperative dialogues found in the FS corpus shows that cooperation is a means to clarify and shorten the dialogue. Thus, maintaining a cooperative attitude is another way of preventing and repairing failures.

#### 3.3.7.1 Additional information

In the following example, A's lack of co-operative behaviour forces B to ask an additional question:

| AFC[I1] | *the two dialoguees have agreed on the outgoing trip* |
|---|---|
| A: | . . . désirez-vous réserver un retour |
| C: | oui c'est à e ça se prend au c d g 2 au deuxième e |

This disrupts the normal thread of the dialogue as the agent introduces a new topic (see section ??) and the caller comes back to the previous one.

In contrast, the following example shows how the agent anticipates the caller's potential follow-on questions.

| BA [2] | |
|---|---|
| A: | ah well you've actually wro(t) rung the wrong number in future you need to dial seven five nine (.7) one eight (.3) one eight (.3) |
| C: | one eight one ei-ght |
| A: | -now let me just look for you |

#### 3.3.7.2 Blocking false inference

Participants expect co-participants to make certain inferences about their aims and motivations. Sometimes a participant signals that one of these inference is not to be drawn. For example in the following extract, after the two participants agreed on a reservation from Paris to Frankfurt, the agent's query about a return reservation, where 'return' re-uses this information, is answered by an affirmative with a contrastive 'but' blocking the expected inference about the return.

| AFC[I88] | |
|---|---|
| A: | oui désirez-vous réserver des retours |
| C: | oui e mais pour cologne paris alors |

Notice the use of the French structure *oui mais ... alors* which makes explicit the correct inference. If the destination and arrival locations were explicit rather than elided, the reply might have been very different.

| | |
|---|---|
| A: | Do you wish a return from Paris to Frankfurt? |
| B: | No, but I do from Nice to Frankfurt |

Similar examples are also found in the German corpus:

| dberl[12] | |
|---|---|
| C: | geht durch |
| A: | wollen Sie weiterfahren oder |
| C: | nein danke bis Nürnberg |

Further work is need to establish how inferences are blocked in the corpora. In the FS the design missed ways of blocking false inferences: for example it would have been better to signal that there are no more seats in economy class when the flight details were presented, instead of after the caller had specified the class that they wished to book.

#### 3.3.7.3 Co-operative reformulation

In some cases the agent's reply may attend to incorrect or problematic features of the caller's request. For example, in the following extract the agent supplies the correct flight number.

| BA[8] | |
|---|---|
| C: | er the bee ay five eight four from turin love |
| A: | five eight four hold on please (15) |
| | er we don't have five eight four sir I think you might mean the five seven nine |

Co-operative answers are tend to be organized according to the following convention: < corrective answer : suggestive answer >.

#### 3.3.7.4 Summarization

Sometimes a request may be answered with a summary of the possible responses rather than a single direct response. For example, in a flight reservations setting there may be several possible flights which satisfy the parameters of the caller's initial request. In such circumstances the agent might respond with a summary such as "We have three flights to Paris tomorrow morning".

| | |
|---|---|
| AF[I41] | |
| A: | oui je vais vous donner les fréquences les jours où il y a des vols directs |
| C: | oui |
| A: | alors les fréquences des vols directs c'est lundi mercredi et vendredi ,,, voulez-vous qu'on regarde le mercredi 2 février e pardon le lundi 2 février |

Here again, the organization is of the form: < summary : suggestive line >.

#### 3.3.7.5 Modulating Outputs

Particularly important information is often delivered at reduced speaking rates compared to the surrounding material. For example, telephone numbers are frequently chunked into groups of 2–4 digits with short pauses in between. Slowing of speech and increasing of amplitude are often used to aid repair after conversational breakdown. One strategy for dealing with breakdown is to switch from normal speech mode to 'spell' mode. There are, of course, many problems in distinguishing spoken letter names but the combination of (slowed, high amplitude) pronunciation and spelling is sufficient to resolve acoustic recognition failures in most cases. In human–human conversation spelling occurs with "informal" rules such as presented above and using international alphabet (alpha, bravo, charly ...). However, in the in FS corpus of human-machine dialogues, callers never used this alphabet.

Lastly, the machine should have notions of its own competences in pronunciation. We have a good illustration in the following extract:

| | |
|---|---|
| FS[159] | |
| C: | je voudrais/, l'NUmŕo du vo:l\, s'il vous plait |
| S: | le numéro de vol est le i j 6737 |
| C: | vous-avez bien dit\,, i j, 735/ |
| S: | le numéro de vol est le i j 6737 |
| C: | vous-avez bien dit, i, j, 7, 3, 5/ |
| S: | le numéro de vol est le i j 6737 |
| C: | #hh hh,, pouvez-vous/, répéter\, s'il vous plait/ |
| S: | le numéro de vol est le i j 6737 |
| C: | vous-avez dit, I, J, 735/ (3) "la ça marche mal" |
| S: | non le numéro de vol est le i j 6737 |
| C: | pouvez-vous, répéter\,,, "j'comprends RIEN du tout h" |
| S: | le numéro de vol est le i j 6 7 3 7 |
| C: | j'suis DÉsolée je n'ai rien compris/, veuillez répéter\ |
| S: | le numéro de vol est le i j 6 7 3 7 |
| C: | "bon/,,ben y a rien a faire\ #hh hh, ta ta ta ta, bon c'est pas gra:ve\ i,, j, 7, 37/, j'suis bouchée hein\ bon/ |
| S: | oui |
| C: | "ah ben voila/,, merci\,,, i, j, 7, 37" (4) "donc", i,j,7, 37, parfait/ |

The caller never understands the flight number, and the system when saying "oui" made an acoustic error (simulated by the accomplice).

In this extract, the caller gave us the way of spelling correctly: include pauses between different kind of information. Thus the correct spelling should have been "i j,, 6 7 3 7" instead of "i j 6 7 3 7" (the error was because "j" and "6" can be confused when the locutor—the synthesis board—is not very good).

## 3.4 Informational structures

### 3.4.1 Reference

#### 3.4.1.1 In information seeking dialogues

**Establishing major discourse objects:** there are two main modes for establishing the major discourse object referenced in the first topic slot of flight enquiries. In the first type, nominal expression provide information to identify uniquely a referential or discourse object. This is the case with flight identifier expressions such as, 'BA 901'. Like proper names, however, it is not always the case that they do uniquely refer: flights with the same destination and arrival locations and times but differing dates of departure may have the same flight number. In most cases the agents do not check the date with the caller: in the face of potential referential ambiguity, it is assumed that the current flight is being talked about. Further work is needed to establish the conditions under which the agent clarifies the date. In the second type, initial lexical information only provides a partial description of the object and its identity is established incrementally.

| BA [2] | | |
|---|---|---|
| 3 | C: | oh good morning hh I wondered I'm |
| 4 | | just checking on a plane that's due in |
| 5 | | tomorrow |
| 6 | A: | ye@s@ |
| 7 | C: | @from@ warsaw |
| 8 | A: | yes |
| 9 | C: | at half past nine on terminal two |
| 10 | | I think k(uh) (.3) I'm just checking |

In the majority of cases the caller volunteers further information. If this is not forthcoming, the agent will prompt the caller for diagnostic information such as the flight number, the arrival or departure times and locations. For example,

| BA [42] | | |
|---|---|---|
| 1 | A: | british airways flight information |
| 2 | | can I help yo-u: |
| 3 | C: | -y*es ur:m: (.) flight (.3) |
| 4 | | from geneva to gatwick |
| 5 | | (.5) |
| 6 | A: | geneva to gatwick do you have a flight |
| 7 | | number ma-dam |
| 8 | C: | -yes seven five nine ¿no ah(m) so- |
| 9 | | (.) seven (.) three (.) seven |
| 10 | A: | seven three seven just a moment please |

Further work will establish what information is diagnostic, what the preferred order of queries is and how these interact with dialogue history (for example, arrival location will not be diagnostic when the caller has already informed the system of the flight's arrival location).

**Alternative descriptions** Once a discourse object has been established, it is referred to by further linguistic expressions which vary in specificity. With referential domains containing more than one possible referent, as the specificity of the expression decreases, so the reference needs to be determined in conjunction with other linguistic and pragmatic clues. Substantial descriptions such as the '903 Frankfurt–London' and definite descriptions like 'this morning's flight from Frankfurt to London' are specific by virtue of their lexical content. Other descriptions, namely indexical expressions such as 'that flight', stand mid-way between substantial descriptions and pronouns: they impose some specific constriants but these constraints are clearly 'indexed' to the dialogue context. At the other extreme, pronouns such as 'it' impose no specific referential constriants and resolving their reference seems to involve using selectional restrictions provided by verbs as well as relevance to the preceding topic and focus.

#### 3.4.1.2 In the reservation task

The major object is of course the reservation itself. Thus, flights are hypothetical during the formulation phase. Afterward, the proposed solution(s) may be referred in different ways.

**Describing the reservation:** if the caller knows exactly the flight in which she wants a seat then she gives precise information: flight number, times, cities... In the FS simulation corpus, the amount and order in which information items are delivered is probably greatly affected by the scenarios, [?]. Another factor is the learning of users: in the beginning of the experimentation callers tend to give all the information present in the scenario, but later they tend to deliver only the crucial information and some of them gave provided parameters one by one waiting for the system to prompt for the information required:

| FS[102] | |
|---|---|
| C: | je désire fai:re-u:ne réservation, d'avion/ sur-un paris-londres s'il vous plait/ |
| S: | paris-londres: quand voulez-vous partir? |
| C: | je désire parti:r le < quatre janvier>,,, 90 |
| S: | paris-londres le 4 janvier: à quelle heure voulez-vous partir? |
| C: | à, SEPT-heures du matin/ |

**Reference in alternatives:** when proposing a bag of solution, the caller may refer to each individual solution with its position in the bag (the first, the second ...) according the order in which they were introduced by the system. However, such "meta-reference" is not the most frequent in human–machine conversation: only 15 occurrences were found in the FS corpus out of 42. Callers tend to refer to the flight with specific coordinates: departur time, arrival time ("le vol de 8 heures", "celui qui part à..." ...).

**Reference of one flight:** if only one flight has been proposed, the only way of referring to the flight is with the use of a neutral pronoun ("il", "ça").

**Other references:** temporal information is either precise (a number) or vague and then introduced by a special preposition ("vers", "aux alentours", ...). Spatial information (namely destinations) are always referred by their name (there were no descriptions like "the capital of France").

### 3.4.2 Focus

It is commonly recognized that there is a connection between focus and pronominalization: the higher the focus level, the less specific the linguistic description. Further work will establish the relations between focus and other notions such as topic as they appear in corpora.

### 3.4.3 Ellipsis

There are countless examples of ellipsis in the corpora. Ellipsis may be classified as linguistic, pragmatic or conversational [?].

Conversational ellipsis are tied to the domain: we say “the company Air France” instead of “the flight company…” or “economy” for “economy class”.

Linguistic ellipsis involves the re-use of information (referential or otherwise) which has been explicitly provided by previous linguistic expressions: the information is recoverable from the (local) context. Elided expressions include verb phrases, prepositional phrases and noun phrases.

For example:

| BA [40] | |
|---|---|
| C: | hello can I check on british airways six three one from athens please (.3) |
| A: | yes I’ll just check [OBJECT] for you hold o:-n |
| C: | -thank you (9.3) |
| A: | [SUBJECT] came in at ten forty this morning (.3) |

The useful context is not only the preceding utterance. For example in FS[98] (see section **??**), the caller’s utterance “et le prix de mon billet” is in continuation of her previous utterance (two turns before) “pouvez-vous me donner l’heure de départ, s’il vous plait”: the perfomative expression “pouvez-vous me donner” is elided. Callers tend to do this very often, in fact they prefer to chunk their requests into smaller ones avoiding requests such as “pouvez-vous me donner l’heure de départ, sur quelle compagnie je dois partir ainsi que le prix de mon billet” although this is correct in written dialogue.

Pragmatic ellipsis involves the (re-)use of information which is implicitly provided by the linguistic expressions in question, and is capable of being made explicit by the addition of other expressions. One obvious example is numbers which may be terminal numbers, flight numbers, clock times or telephone numbers:

| BA [1] | | |
|---|---|---|
| 20 | C: | and which terminal |
| 21 | A: | (sorry number) one |

### 3.4.4 Topics and Topic Shifting

Dialogues which have a predictable overall organization, such as those within the telephone information service domain, are monotopical: while there may be more than one topic addressed, the participants orient to the expectation of a single topic. The main topic of the dialogue is established in the first topic slot which follows the opening section. This topic imposes constraints on the subsequent dialogue; for example, reference is highly attuned to this topic.

There is a noticeable preference for further topics to be semantically linked, or relevant, to the preceding topic: new topics tend to be relevant to the preceding topic of the dialogue. For example, topics which are relevant to a request for the arrival time of a flight include the arrival time and place. However, some topics are not linked to the preceding topic, especially if it is complete and occurs immediately prior to a pre-closing sequence. Most of these are signalled by markers such as increased amplitude, hesitancy as well as explicit markers of discontinuity such as 'hey' and 'right'.

| BA [5] | | |
|---|---|---|
| 39 | C: | rig-ht can you |
| 40 | A: | -yeh |
| 41 | C: | also answer another question I'm |
| 42 | | coming back into gatwi-ck |

| BA [7] | | |
|---|---|---|
| 16 | C: | right thank you very much¿are you able |
| 17 | | to tell me (.5) er flight arrival times from |
| 18 | | zimbabwe oh no you can't i- its its at |
| 19 | | gatwick |

While both agent and caller initiate linked topic shifts, there is a marked preference for only the caller to initiate unlinked shifts. Furthermore, there are also unlinked topic shifts which are not marked; for example, when the caller talks to someone other than the agent, and caller digressions about who they know on a flight, etc[6].

There may be examples of flashforwards although we have yet to find examples of them in the corpora. A few flashbacks have been found in the BA corpus.

## 3.5 Prosodic phenomena in dialogue

Prosodic phenomena are often less amenable to analysis than text-based linguistic phenomena. Nevertheless, it is generally agreed to have as acoustic correlates pitch, duration and intensity; many insightful analyses have concentrated on pitch (described at a phonological level as *intonation*) alone. Intonation comprises several phenomena thought to have linguistic significance:

- *phrasing*, indicating how material is broken up within an utterance. This may be used for example to effect syntactic disambiguation, or to indicate chunking of material, as in lists;
- *accent placement*: the intonation of a language such as English may be described as a sequence of accents, for which the prominence is carried

[6]Digressions could perhaps been seen as topic shifts which are not oriented to the main topic of the call.

by marked obtrusion of pitch, or pitch movement. Typically such a pitch prominence is associated with the syllable of a word carrying lexical stress, so that that word, and possibly the constituent(s) to which it belongs, is prosodically focussed. Flexibility of accent placement differs across languages. English, and German, can achieve *narrow*, or 'contrastive' focus, by accenting a certain word at the expense of its neighbours; Italian, and to a lesser extent French, will tend to rely more on word-order to effect selective highlighting;

- *tune assignment*: techniques for describing intonational melodies vary; descriptions for English are largely based on the 'tone sequence' model [?], which divides the melody up into a sequence of tune-carrying pitch accent units, with some principle of spread or interpolation affecting the intervening contour. We may therefore speak of the local contours associated with accents; the final local contour in a phrase is referred to as the *nucleus*, and is generally considered to carry the most significant functional load. The associated movement may, for example, determine how an utterance is intended to be interpreted, as a dialogue act, by a hearer;
- *accent scaling*: the peaks and valleys an accent reach with respect to its neighbours, may to some extent be considered separately from the phenomena of tune. The effect may be one of heightened excitement, insistence, emphasis, contrast;
- *pitch register setting*: this property tends to be applied to entire phrases: whether they are within a normal pitch range for the speaker, or relatively high or low. This may affect topic, and rhetorical coherence.

In the remainder of this section, we shall indicate typical dialogue phenomena which are wholly or partly cued prosodically.

**Greetings, farewells, politeness phenomena** These tend to have stylised, (chanting, stepping) tunes associated with them:

```
~~=flight infor-_mation | ~~=british -_airways  | =good day
-_can I _-help you
```

The reader may like to verify for him/herself that a non-stylised contour in the context of a greeting may sound peremptory, or rude. It may be because they tend to be associated with phatic utterances, that such contours when used say with the delivery of information, impart a mechanical, ritualistic quality to such a delivery; this may be associated with boredom on the part of the speaker. However it may also be a component of an exaggerated, ‘commercial’ voice.

One melodic phenomenon associated with deferential politeness, is the temporary adoption of a high, almost falsetto, register setting. This is reported to occur in many languages. In English it is common as part of the formulation of requests:

[*falsetto* I wonder if you could tell me the] times of trains ...

It does not persist into the information-bearing part of the utterance.

**Chunking** We use this term to refer to the division of utterances, via phrasing, into manageable-sized meaning-bearing units. In the specific case of a large quantity of information being delivered, in the form of a ‘list’, phrase boundaries between units may be marked by the intonational equivalent of commas. In English, German and French these boundaries are often marked by a tone with a rising component; only the final phrase will have a nucleus with an associated global communicative function. Such boundaries may be treated as intervention points by the listener, for instance to demand clarification, repetition, or feedback, as in the following example:

```
[16] T1:SB:877 (T)
18 A: yes iberia's number uh(p) still searching
19 here we go (2.5) ^oh ;/one
20 C: /yeah
21 A: =eight ;=nine !!\/seven
22 C: =eight nine /seven
23 (.)
24 A: =seven \/nine
25 C: =seven /nine
26 (.)
27 A: =four ;\_one
28 C: four one -thank (you)
```

It seems though that the listener needs additional cues, as well as intonationally marked chunk boundaries; a non-filled pause by the speaker at such a boundary will serve as a suitable turn-giving cue.

It will be observed in the above example that English speakers often use a fall-rise tone to indicate boundaries between chunks. A similar phenomena has been observed in data from the French simulations:

*Je voudrais réserver un aller Paris-Hyères \ /,,, pour le mardi 8 janvier \ / ,,, départ 21 heures vingt cinq \ / ,,, arrivée 22 heures quarante cinq \ / ,,, par le vol, it 6455 \ /*

However the use of this tone is sufficiently rare, and is considered by French informants to be stylised. More commonly it has been observed that French speakers use pauses, together with level or rising tones, to mark chunk boundaries.

**Marking discourse organisation** We have seen how 'list', or *continuation* melodies, may be used to indicate that a speaker's utterance is not complete. We may also note the use of (sub-)topic opening and closing devices. In English, an 'intonational paragraph', or *paratone*[?], is often associated with discourse topic. This is frequently opened with on a high pitch register, or *key*; in subsequent intonation phrases the key moves downwards, and the paratone may close with a low fall. This phenomenon is more readily observed in monologue than in dialogue; some researchers building text-to-speech systems have used knowledge of hierarchical discourse organisation to plan corresponding tonal hierarchies [?]. What we observe in conversation is complicated by the interplay of parties; we may associate the 'deferential falsetto' discussed earlier with topic opening, since it is often used in opening requests, but it is not clear that these always coincide. On the other hand, there is evidence that speakers associate a lowish fall with closing, either of a list utterance, a local exchange, a topic, or the entire conversation. In these latter cases it is often associated with a discourse marker whose function is closing, such as "right", "okay".

In the following example, the caller's problem formulation is terminated with a fall-to-low:

```
I wonder if you could tell me
I ^=haven't actually got the \/flight number |
but there =should be a flight coming in
from \/crete this morning (.3)| and I'm
=wondering what time it \_arrives
```

In this example, the information provider has been unable to help the caller, and gives a number as second best:

```
T1:SB:9588 (T)
A: I think the ^^\best thing for you to /do is |
        ring gatwick on =oh ;=two nine /three
```

```
C: oh two nine three
A: two double eight
C:  two double eight
A: double -_two
C: \_double two \_lovel-y
```

Here A's utterance has the shape of an intonational paragraph, or 'paratone', with high onset on "best", falling to an expected low-fall nucleus on "double two". In fact A softens this by using a stylised tone to close on. It is left to B to close with a double low fall.

**Intonational focussing** We have noted that the pragmatic effects of accent placement are probably more marked for English and German. In many languages, the final, most prominent pitch-accented syllable, or nucleus, tends to be on the last noun phrase. De-accenting material which would normally be accented, and the converse, accenting normally unaccented material, achieves the effect of highlighting the information associated with the latter, or playing down information associated with the former. In the following example, the two parties are now aware that the flight in question is not British Airways:

```
C: (hu-) =what airline !!\did you think
it was
```

In this utterance, "did" receives *narrow focus*—sometimes rather misleadingly described as 'contrastive focus'—at the expense of the rest of the tone group. There is no new information here, so the emphasis on "did" must either serve to highlight the past tense aspect, pointing to the common knowledge that A has been disabused of her former belief, or it could be taken to mean "what airline did you (in truth) think it was", perhaps accusing the caller of using the wrong service. The following exchange demonstrates the cumulative effects of deaccenting old information, together with the use of narrow focus to achieve contrastive highlighting:

```
[19] T1:SB:1413 (T)
15 A:                       ^=nine ;=forty ;\_five
16 C: (huh-) nine forty five- (       )
17 A:                        - \yes | it ^\/was
18 scheduled at ;\_seven thirty five | it ^\/actually
19 landed at ;\_nine forty fi:ve
```

Speaker A treats "thirty five" as close enough to "forty five" to count as old information—to do otherwise would detract from the first contrastive accent, on seven. The presence of a second contrastive accent, on “nine”, demonstrates that what matters is not that material may have been mentioned previously, but in the words of Bolinger, that material is of ‘interest’.

One phenomenon deserving mention alongside selective focus, is the assignment of a so-called ‘referring’ accent to material which is referenced, in order to make some new point. In this example, both speakers have initially made the false presupposition that there are shuttle flights to aberdeen.

```
[23] T1:SB:1608 (T)
8 ^that’s not \shuttle | ^\not aber\/deen
9 C: oh isn’t i:t
10 A: it’s a =regular: bookable \service aber/deen
```

The referring accent is usually associated with a fall-rise tone [7], whereas the new material carries a tone which conveys the global communicative function (in this case, assertion).

It should also be noted that attempts to account for pitch accent wholly in terms of information focus are doomed to failure—as Halliday points out, the matter is one of speaker choice, or in Bolinger’s phrase “Accent is predictable—if you’re a mind-reader”. It is often the case, for example, that normally unaccented words receive pitch accents principally for the sake of euphony, as in the following example, where there seems little reason for putting the nuclear accent on “for”, except that to put it on “something” or “you” would have given undesired prominence to those words, both of which are associated with ‘old’ information.

```
[22] T1:SB:1552 (T)
16 about it |? =maybe I can: |? =try and find \/out
17 someth-ing !!\for you=
```

One analysis of the accent on “for” would thus describe it as a *default accent.* An alternative analysis, closer to Bolinger’s position, would be that although “for” is a function word, it is not semantically empty, and that what is raised to prominence here is the property of being beneficiary.

*Speech Rate* is also a possible resource that speakers can use to give additional weight to parts of an utterance. For example, a slow rate has been found in the French simulations corpus to be used by speakers when there was perceived uncertainty about the intelligibility of some parameter.

[7]In the example, assuming that “aberdeen” receives a referring accent in both cases, the first is a fall-rise, while the second is a simple rise. In fact a fall-rise would be possible also here, but less easy to articulate after a fall. It may be that some ‘tonal linking’ is taking place here.

**Conveying Intention or Attitude** There are no simple context-independent associations between syntactic features and dialogue acts, that are exceptionless. In attempting to match contours to dialogue acts, we come across a similar problem. We have noted that in many languages a falling tone, or a low tone, is associated with finality, or assertion, whereas a rising or non-low tone is associated with suspension, or question, uncertainty. However it is by no means always the case, for example, that a syntactically unmarked 'question' is capable of being detected as such via intonation. A fall-to-low does seem often to be associated with information giving, for example:

```
[21] T1:SB:1466 (T)
12 ^-_thirteen ;\_thirty
```

—where it seems to convey a note of confident authority. This may however be most the case in dialogues where it is clear that the caller only requires a limited amount of information, and is probably busy. If the information provider expects further requests to be made, or wishes to prolong the dialogue for some reason, it is perfectly in order to finish a turn with a rising nucleus. Because a fall to low signals finality, a potential conflict arises when the speaker needs to use it to conclude a chunked sequence, but does not wish to close the conversation, as in this example, part of which has already been quoted:

```
[16] T1:SB:877 (T)
24 A: =seven \/nine
25 C: =seven /nine
26 (.)
27 A: =four ;\_one
28 C: ~=four ;\one | ~\_thank (you)
29 A:                now did you say it was the
30 eye bee six one four
```

C takes A's "four one" as beginning a closing sequence. In order to abort this pre-closing sequence, A has to rush in, overlapping C.

One phenomena where contour might be expected to give a clear signal of speaker's intention, is that of echoes. These may either be intended as 'backchannel signals', not expecting a response, or as requests for confirmation or clarification. In the first case we would expect low falls, whereas requests would be expected to be marked by rising intonation. In fact, as the example above demonstrates, context-independent generalisations are dangerous: giving feedback during a chunked sequence, rising intonation may be used. It is nevertheless the case that speakers often wish to disambiguate mere feedback from request utterances, and will tend to use a softer voice and fall to low to do this. The discriminatory power of rising/falling contours to distinguish the function of echoic utterances has also been observed for German. Empirical observations for German, based on a corpus of 1999 test utterances, have additionally validated the function of rising intonation in general to indicate questions.

One further phenomenon is worth mentioning: the use, in English, of a fall-rise tone to convey uncertainty [?]:

```
[8] T1:SB:344 (T)
10 A: five eight four hold on please
11 (15)
12 er we ^=don't have a =five eight \/four sir
13 I ^=think you \/might mean |? the =five seven \nine
```

Use of this tone often has a conciliatory, polite softening effect. For this reason, it is particularly appropriate in cases of corrective or suggestive utterances, where results cannot be found that exactly match the client's stipulation.

## 3.6 Conclusion

This chapter has given an overview of the complexity of oral dialogues. The intention has been to describe as many as possible of the *observable* phenomena seen in both human-human and human-machine dialogues, as far as possible in a theory neutral fashion. Ideally the Sundial system should be able to recognise and support all these phenomena; in practice, of course, this is an unrealisable goal and some selection has to be made. The phenomena to be supported need to be chosen with two criteria in mind: firstly, those which are common or crucial to effective communication should be prefered over those which are rare or incidental; and, secondly, phenomena which are relatively easy to handle given the current state of the art should be preferred over those which cannot be implemented without a great deal of further research.

The next chapter therefore reviews the phenomena described here to examine how they might be modelled computationally in a dialogue manager. At the beginning of the following chapter, a list of phenomena which it would be reasonable to handle in the first Sundial demonstrator system is identified.

# Chapter 4

# Approaches to handling dialogue phenomena

## 4.1 Introduction and State of the Art

In the previous chapter we have seen an analysis of dialogue phenomena from the pre-theoretic viewpoints of Conversational Analysis and Linguistics. Human dialogue exhibits a wide range of phenomena which yield only reluctantly to formal treatment. There are a number of major difficulties:

1. The units of dialogic organisation: 'speech acts', 'dialogue acts', 'illocutionary acts'—to take a handful of terms which are more or less synonymous in the literature—are not easily definable (see chapter one), in contrast say to units of linguistic organisation;
2. Again unlike linguistic organisation, dialogic well-formedness may be impossible to formulate, since speakers continually flaunt any conventions the observer may wish to impose, such as adjacency pairing; against this we may certainly set the notion of 'preferred' sequences, and design man-machine information dialogues with these in mind;
3. Although *post hoc* analyses may throw up revealing patterns, the future course of a dialogue for one of the speakers is highly non-deterministic, unless the relation between speakers is such that one is dominant and able to dictate that course;
4. Lastly, we may observe that speakers' cues to the dialogic status of their utterances are often minimal, and ambiguous.

Faced with such difficulties, philosophers of language and computational linguists have sought to increase the explanatory power of their descriptive apparatus by recourse to underlying, implicitly present contextual representations and mechanisms. For example, non-determinism may lose prominence when dialogue is viewed from the level of the intentionality of the participants [**?**, **?**] . On the other hand, for the builders of speech dialogue systems, the option of directed dialogue, in which the system takes a dominant role, may be highly attractive—even when tempered with considerations of usability and friendliness. Another conclusion, that man-machine systems need to eschew the enigmatic terseness of human dialogue, and be painstakingly explicit in its cues, may be set against the potential drawbacks of unnaturalness, and excessive transaction time.

In describing previous work in the field, we focus on three major areas:

1. Work describing underlying patterns of intentionality;
2. More 'engineering' approaches, aimed at producing efficient systems with well understood dialogue design;
3. Work aimed at modelling context dependence, for example that exhibited by ellipsis, anaphora, and by discourse topic phenomena.

### 4.1.1 Modelling intentionality

Cohen [**?**] applied AI planning formalisms to the description and implementation of speech acts in computer programs. He proposed that conversation be viewed as a sequence of actions, in which the participants affect each others' beliefs and goals, via speech. Such a sequence could be extended to include nonlinguistic actions aimed at fulfilling an agent's intentions, such as getting a door opened. In a complementary approach, Allen [**?**, **?**] produced a computer program in which the system was concerned with modelling the user's underlying plans and goals, using the evidence of speech acts produced. Allen used such a *plan inference* system to provide a rational account of some phenomena in cooperative dialogue, as in the following exchange:

(1) patron: when does the Windsor train leave?
clerk: 3:15, at gate 7.

Here the clerk provides, gratuitously, unsought-for information. Allen suggested this was because s/he is able to infer that the patron has the plan of catching the Windsor train; on reconstruction of such a plan, the clerk may detect that the patron needs to leave from a certain gate, and may not know which. The additional information is designed to circumvent this potential obstacle. The mechanism used by Allen depends on a set of *plan inference rules*, which are essentially inverse to plan construction rules used in top-down plan expansion. For example, if the speaker is “S”, the interlocutor “A”, and “SBAW” is to be read “speaker believes hearer wants”, then we have (among others) the following rules:

SBAW(A knowref D) $\subset_i$ SBAW(P(D))
SBAW(Precond) $\subset_i$ SBAW(Act) — where *Precond* is a precondition of *Act*

The first states that if the speaker thinks A wants to know the referent of D, then he wants some condition of which D is a parameter to come about. An example of this is the one given. The patron needs the time of the Windsor train because knowing this means knowing a parameter of a plan to catch the train. It is clearly not always possible to interpret unambiguously a speaker’s underlying intention, and the plan inference rules do not represent necessary entailments, but rather, possible ones. Being able to open the door is a precondition to performing this action, so the hearer of:

(2) Can you open the door

should be able to apply the second plan inference rule, and derive the underlying request. However such an interpretation of the indirect speech act is defeasible—a literal interpretation is possible, if unlikely. In searching for underlying plans Allen’s system is guided by plausible expectations about agents’ goals, and a set of *rating heuristics*, which serve to order candidate hypotheses.

Allen’s treatment of speech acts, as evidence of underlying plans, has been immensely influential in computational linguistics, providing as it does an alternative to earlier syntax-based accounts. Litman (Litman and Allen 1985) considered the case of *clarification subdialogues* as a special case of obstacle detection. If an agent is trying to reconstruct his interlocutor’s plan, and finds some information missing, he may plan for a clarification question. In order for her system to reason about plans, Litman included the latter as objects in her ontology, and allowed for *metaplans*, such as SEEK-ID-PARAMETER, which came into action when a parameter of some plan was not known. Instances of plans were kept on a stack, so that when the purposes of a plan were achieved, it could be popped from the stack. The input speech act could be used to continue the current plan inference, in one of three ways:

1. continuing the top (suspended) plan on the stack;

2. introducing a clarification metaplan, for some plan on the stack;
3. constructing a new plan

Like Allen, Litman used domain-specific expectations of speakers' goals.

### 4.1.2 Interpreting ellipsis

A further application of plan recognition has been the in the interpretation of sentence fragments. Carberry [**?**, **?**] remarked that a range of pragmatic inferences may be associated with these; compare:

(3) A: The Korean jet shot down by the Soviets was a spy plane
B1: With 269 people on board?
B2: With infrared cameras on board?

While B2 is concerned with obtaining (or confirming) additional information, B1 expresses doubt about the underlying proposition. Carberry poses the question: what knowledge does an information provider need, to cope with such fragmentary utterances? She identifies four major components to this:

- a task-related plan, sufficiently explicit to account for the information seeker's goals as revealed in the previous discourse;
- a representation of shared beliefs;
- a set of anticipated discourse goals, based on general conversational principles;
- processing knowledge, including plan-recognition strategies and focussing mechanisms.

Processing fragments takes place as follows. A discourse component suggests discourse goals that the information seeker IS might be pursuing. These are ordered in stack fashion, and serve to schedule responses to IS's utterances in such a way that conversational principles are conformed with. For example, having asked a question, the information provider may push onto the stack the discourse expectation "Answer Question". Carberry provides rules relating each discourse expectation to a family of possible discourse goals, which IS might use to satisfy that expectation. Plan recognition techniques, similar in essence to those of Allen, are used to propose possible attachment points for IS's incoming utterances, to a *context model* representing that portion of IS's plan derived so far. Conflicts between alternative choices are resolved using the possible discourse goals, together with a focussing mechanism that attempts to enforce local coherence by preferring associations in or close to the area identified in the preceding discourse as the current focus of attention. Focussing is discussed further in Section **??** below.

Complementary to the work of Cohen and Allen, primarily concerned with modelling just one agent, is that of Power [**?**]. He proposed a computational theory of communication which models both agents. In his system, two agents are initially equipped with possibly disparate sets of belief and goals, in a 'world' consisting simply of the two robots themselves, a door and a bolt, and simple 'laws of nature' governing action in that world. The robots, which are simply two identical programs set up with different initial conditions, have as resources the ability to plan privately, and the ability to initiate and participate in *conversational procedures*. These are used as a basis for conversational structure, and may themselves call embedded conversational procedures. An example:

CONVERSATIONAL PROCEDURE ASK(S1, S2, Q)

1. S1 composes a sentence U which expresses Q as a question, and utters it.
2. S2 reads U and obtains a value for Q. He records that S1 cannot see the object mentioned in Q, and then inspects his world model to see if Q is true. If he finds no information there he says I DON'T KNOW, otherwise YES or NO as appropriate.
3. S1 reads S2's reply. If it is YES or NO he updates his world model appropriately. If it is I DON'T KNOW he records that S2 cannot see the object mentioned in Q.

Both speakers share the same inventory of conversational procedures, and each of these is intended to achieve a given purpose for the initiator. For example the conversational procedure ASK is initiated by a speaker whose purpose is to obtain information. Then in order to get the addressee to know which conversational procedure he is expected to take a role in, the initiatiator has to announce it, as for example, for ASK:

May I ask you a question?

In the initial stage of a conversation, the robots, being cooperative, agree to pursue a common goal, such as that of robot1 being on the same side of the door as robot2. The robots develop in parallel identical plans, planning in private, but identifying where necessary those parts of the planning tree which can only be accomplished by a single robot. For these, conversational procedures are used, both to accomplish delegation of responsibility and to announce its results. For example, to achieve the goal of having the door open, one agent may be assigned the responsibility of pushing it, and later inform the other agent of the new state of affairs. Allowing planning and the execution of plans to interact, as well as the modelling of two agents, marks Power's work as different from the plan-recognition work considered so far.

In Power's system the initiator's choice of conversational procedure is dictated by a low-level action in his plan tree. By contrast, in the work of Houghton [?], based closely on that of Power, a speaker simply has a goal to achieve, and possibly a number of options open. Conversational procedures, which Houghton calls *interaction frames* are chosen by a speaker on the basis of reasoning about their effects on the interlocutors knowledge and beliefs, more as in Cohen's and Allen's work. Searlian-like preconditions apply to interaction frames, prior to their initiation, and a speaker knows what effect successful joint performance of an interaction frame will have. For example, Houghton's interaction frame for FIND OUT, has as 'precondition' that the question is worth asking, in the sense that there is a chance that the addressee may know the answer, and an effect on successful completion to the effect that the initiator will know what he needed to know. Interaction frames assign parts to the speakers, and for each turn of each speaker, initialise a message structure to serve as input to a separate linguistic generation component.

Wilensky [?] argues that the kinds of commonsense plan underlying language understanding and production are more complex than those imported from the AI planning paradigm. They take no account of interactions between possibly incompatible goals, for example, the interaction between the immediate goal of getting more disc space, and the 'preservation goal' of not losing existing files; or the need for metaplanning, for example, in trying to create plans which will be economical. He suggests the powerful mechanism of *projection*—whereby a proposed plan is simulated in a hypothetical world model, revealing flaws, inconsistencies, unforeseen (and undesirable) states of affairs. Wilensky's general algorithm for relating new inputs to context is similar to Allen's insofar as it attempts to chain both upwards from observed actions to hypothesized goals, and downwards from presumed or stated goals to actions. However, after the input has been explained, a check is made to determine if there are any associations between it and a subsequent part of the plan: if these exist, they are recorded as 'predictions', and used in explaining later input. Wilensky observes that when the situation degrades into one in which all future inputs are completely accounted for by predictions, we have a case of script application.

### 4.1.3 Engineering approaches to dialogue

A dialogue system in which the system or user takes the dominant role may be simply an operating system, or a database front end. Front ends to databases have been a major application of natural language technology, since the BBN LUNAR system [?]. Such systems may embody sophisticated translation into query languages, and may even model context to the extent that anaphoric and elliptical utterances are possible. These are interesting for their sophistication of natural language technology, but are of little interest as dialogue systems.

Of more significance to our research are those systems such as GUS [?], in which some degree of mixed initiative was allowed. In GUS (a text-input dialogue system in the flight reservations domain) this was effected by assuming that the user's utterances were relevant to the task, and using unsolicited inputs to fill in parameter slots in datastructures representing the task; failing this however it was a simple matter for the system to take the initiative, by tracing through the task parameters in a systematic manner and requesting those that it did not have a value for.

A related system that was speech-based was VODIS [?]. This used a similar mechanism of filling in a datastructure representing the task, to allow some caller initiative, with a default of system initiative. Besides operating with speech input and output, it possessed a number of other interesting characteristics:

1. The naturalness of the dialogue was enhanced by following general patterns that specialised according to task context;
2. The dialogue manager was able to predictively constrain the recogniser, by basing predictions on the current subtask. However the system was able to encompass not only input directly relevant to the current subtask, but also that with a wider scope;
3. A flexible confirmation strategy was used;
4. The system had some awareness of current transaction success, and was able to modify its strategy to a much more conservative and directed one, in cases when this was poor.

Both of the two mixed-initiative systems outlined above depend heavily on knowledge structured around the nature of the task; dialogic knowledge as such is at best implicit. Other systems have sought to model conversational organisation, by the use of some form of dialogue rules. One popular approach has been to use a form of ATN, where the nodes represent states, and arcs conversational moves. An embedded subdialogue can if necessary be represented by a 'PUSH' arc [?]. Equivalent, but apparently more declarative, are the LOKI dialogue rules, which are formulated as an augmented context free grammar [?]. These systems are capable of emulating complex patterns of dialogic organisation, from greeting sequences to adjacency pairs to topic organisation. However they are not especially task-directed, and therefore may have difficulty in dealing with user initiatives.

### 4.1.4 Modelling context dependence

The principle that that focus and discourse topic organisation may be task-dependent was formulated by Grosz [?, ?]. She analysed task-oriented expert-apprentice dialogues, and found that the set of possible antecedents to a referring expression such as "the screw" depended on the current subtask focussed on in the dialogue. This meant that if a task was returned to after an embedded sub-task had been dealt with, a referring expression could refer back to an earlier, not recency mentioned antecedent. This was in contrast to earlier work, for example Winograd [?], which assumed that antecedents were not structured, but ordered linearly according to recency of mention, so far as their anaphoric potential was concerned.

Grosz used the terminology 'focus space' to describe a set of entities in the domain model which at a given time in the traversal of the task space were particularly accessible. The current focus space was termed 'active'; those above it in the task hierarchy, 'open' focus spaces. Interpretation of a referring expression was handled by searching for an antecedent in the active and open spaces. If this proved unsuccessful then a focus shift was assumed to have happened, and neighbouring spaces were searched, and the active and open spaces reset if necessary.

### 4.1.5 'Local focus'

The principle of mutual dependence of focussing and anaphor resolution also formed part of the work of Sidner [?]. However rather than being concerned with explicating the global structure of discourse, Sidner examined focussing as a further constraint to sit beside syntactic and semantic/inferential constraints in cases of referential ambiguity, especially when pronouns such as "it", "him", "she" were being used. Anaphoric coherence is maintained by keeping track of foci (rementioned items) and potential foci (items recently mentioned). Foci can be *actor foci*, if their antecedents were agents, or *discourse foci* otherwise. In both cases, a stack is maintained of past foci—resolution with respect to a focus earlier on the stack results in the upper portions of the stack being discarded.

This is consonant with Grosz's hierarchical view of focus spaces, and allows foci temporarily present during embedded parts of the discourse to disappear when earlier parts are returned to.

A simpler account of local anaphoric coherence is given by Grosz Joshi and Weinstein [?]. In this the term *forward looking centre* corresponds to Sidner's potential discourse focus, and is used to refer to those entities mentioned in the current utterance, but not necessarily established as foci; the term *backward looking centre* corresponds to Sidner's discourse focus, and refers to an entity mentioned in the present and the previous utterance. The authors do not distinguish between actor and discourse focus, and do not use a focus stack, this being performed by the global structure of focus spaces relating to task structure, as in Grosz's earlier work.

More recently, Grosz and Sidner [?], [?] have proposed a schematic framework for relating the structure that underlies anaphoric felicity (which they rename *attentional structure*), the structure imposed by underlying tasks and goals (the *intentional structure*), and the surface organisation of discourse. The latter, which is a hierarchical structure with nodes corresponding to discourse segments (or *discourse segment purposes*) and the root corresponding to the purpose of the whole discourse, is what attentional structure derives from. So far this is congruent with Grosz's notion of focus spaces tracking the underlying task. However they make the point that intentional structure is not simply structured according to dominance (lower level purposes contribute to higher level ones), but that in many cases, *satisfaction-precedence* has a part to play—ie, some intentions need to be satisfied before others. They consider interruptions, in which the suspended thread of discourse may be saved up on a stack until returned to, as in:

(4) John came by
and left the groceries
*Stop that*
*you kids*
and I put them away
after he left

—where "them" can only refer to the groceries—yet nevertheless, Grosz and Sidner argue, the piece of intentional structure corresponding to the interruption is entirely independent of that corresponding to the main narrative, and is therefore erroneously described as being embedded in it. They conclude that attentional structure is not a simple derivative of intentional structure. Further evidence comes from the case of flashbacks. A speaker may recall a discourse segment purpose which satisfaction-precedes part of the current one, and therefore flash back to it, before continuing where he left off. The relationship between the two pieces of intentional structure is therefore not one of dominance, despite the corresponding focus spaces being stacked.

In addition, Grosz and Sidner examine the use of forward and backward centres, as a mechanism for describing more local anaphoric behaviour, as may be necessary in the case of pronouns. They point out that in contrast to the discourse segment purpose, which remains constant for the discourse surface, the backward centre may shift, allowing different entities to become more salient in the course of the segment. They suggest a reason for the confusion in the literature concerning the term 'topic'—what writers refer to as 'sentence topic' corresponds to the backward looking centre, whereas 'discourse topic' corresponds to the discourse segment purpose.

### 4.1.6 Overview of this chapter

In the remainder of this chapter, we review some of the phenomena discussed in chapter three, and provide pointers to how a dialogue system might handle them; we discuss design principles, and issues pertaining to Knowledge Representation. We discuss dialogue 'histories', those dynamic registers which maintain some kind of trace of the past utterances and their interpretation, and hence provide sufficient context for the interpretation and generation of future utterances.

## 4.2 Handling observed dialogue phenomena

### 4.2.1 Openings and closings

Human protocols for organising conversation are well-understood and therefore an attempt should be made to use them in human-computer dialogue. Dialogue phenomena in the SUNDIAL system are managed by a component expert in dialogic interaction. Since the system inevitably takes the initiative with openings, these are relatively unproblematic, and may be hard-wired into the dialogue rules. Closings are potentially a source of problem. The cues that speakers give to indicate their wish to close may be more or less explicit—dialogue strategies need to be devised whereby the system always gives explicit cues—such as asking "can I give you any further information"—and only acts on those cues of the caller which are explicit or have been confirmed. The system must also be prepared to deal with callers who do not respect conversational norms when talking to a machine and hang up unexpectedly.

### 4.2.2 Request formulation

Request formulations are potentially problematic for speech recognition systems, being a) long and b) largely unpredictable. One strategy successfully used in speech dialogue systems (eg VODIS [?]) is for the system to prompt narrowly, so as to elicit a manageable portion of task information at a time. If the caller's initial request formulation contains too much information, much of it can be ignored, and elicited subsequently. Such a strategy has proved successful in the simulations for the Flight Reservations task.

In the SUNDIAL architecture, a task component is responsible for the acquisition of task parameters. If results from simulation and human-human corpora indicate a preferred order of giving task parameters, these preferences may be used as a default ordering, and may even be used predictively to guide the recogniser in initial stages. However, unless the dialogue is narrowly guided, it would be a mistake to rule out non-standard orderings.

A problem arises concerning how much information is needed by the system, in order to attempt a request resolution. In the flight enquiries domain, a flight number is usually sufficient; if this is not known, a number of parameters will be needed. It is necessary that knowledge about tasks should include specificity heuristics, which may be used to determine if the currently requested task is sufficiently specific, and if not trigger a request for further parameters.

### 4.2.3 Request resolution

We may distinguish two phases in request resolution: presentation of information and acting on the caller's choices. In applications that are only concerned with enquiries, the latter phase does not take place.

Presentation of information provides a number of interesting challenges for a message output subsystem. Caller's requests may be deliberately vague: to attempt to narrow them sufficiently so that only one database record at a time is retrieved is unnatural, and inefficient in information-transfer terms. The system must therefore organise output corresponding to a number of records. This may be done by discovering generalities in the output, so presenting the user with a browsing facility. Such work requires cooperation between message output, dialogue management, and task management. Alternatively, if the number of records is manageable, it may be simpler to output all the data. At message planning time the data may be divided into chunks; again dialogue management needs to be informed about the possibility of user interventions.

Request resolution concerned with the system taking some action poses less problems of a technical nature, but a number of strategic problems, such as the need for security, account numbers, precautionary validation. This suggests that the representation of tasks is only partly declarative, but must also be procedural and sequential.

### 4.2.4 Adjacency pairs and embedded exchanges

Adjacency pair organisation, although defeasible, provides a powerful predictive and cohesive device, in the design of dialogue rules. In a normal mode of interaction, requests for information may be expected to be satisfied in adjacent or near-adjacent turns. When this breaks down, remedial action may be taken. The phenomena of embedded exchanges comes about when some goal is suspended while a goal with a higher priority, such as a confirmation or clarification, is carried out. The system of dialogic rules, which encapsulate the normativeness of adjacency pairs, and allow for embedded exchanges, is discussed in sections **??** and **??**.

### 4.2.5 Failure and Repair

We may distinguish between failure on the part of the user, and system failure:

1. User failure may occur because the user's utterance is ill-formed at one of a number of levels: phonetic, lexical, syntactic, semantic. We shall consider errors from the syntactic level upwards:

    - at the syntactic level, the system may not be able to find a parse which spans the period of the utterance. This at least would be the situation with parsing a string of text. When parsing a lattice of word hypotheses, the situation is complicated, because there may be several permissible parses, but because of some irregularity in its formation, the speaker's utterance is not one of them. In practice such a situation may only be detected if the permissible alternatives are ruled out, either by strong predictions or by subsequent filtering. In such a case it may be possible to relax syntactic constraints locally, in order to detect the ill-formed input. Alternatively, the grammar may be augmented with meta-rules to filter out ill-formedness such as false-starts and repetitions, or filled hesitations, should these be detected by lexical access. Such rules however may lead to a considerable decrease in parser efficiency.

  - semantic incoherence too may in principle be detected during parsing, but again this may only happen if other permissible alternatives can be ruled out. In the latter case, some relaxation of local constraints will be necessary if repair is to be effected.
  - incoherence at other levels may in principle be detected by the dialogue manager. However if interpretation has reached this stage, an interactive repair, telling the user more or less specifically that failure has been detected, and inviting self-repair, becomes possible.
  - of particular interest is the class of failures which result from false presuppositions about the domain on the part of the user. We shall consider these in more detail in section **??**.

2. System failures may come about because the system's static knowledge is incomplete, or because algorithmic choices, and thresholds, lead to lack of completeness, resulting in the system being unable to cope with what the caller may consider to be perfectly well-formed input [1]. In such cases, the assumption that the input is ill-formed would be invalid, as would be approaches which attempted to correct it using some form of local constraint relaxation. On the other hand, as was the case with caller failure, the system may seek repair interactively.

The above account is somewhat idealised. In practice, the system may be no more successful than say a compiler, in locating the source of error. More significantly, it may not be possible to determine whether the failure was the fault of the system or the caller. Incorrect diagnosis may lead to a fruitless waste of system resources; on the other hand, if failure is to be dealt with by a dialogic repair sequence, it is important that the system should not assign blame, since if this is done incorrectly, confidence will be lost.

### 4.2.6 Cooperative answers

When some request to the application fails, a cooperative system will not simply signal failure, but may give the user some indication of the reasons for failure, and suggest options which could lead to success. These two types of response are termed *corrective* and *suggestive* [**?**, **?**]. They may be handled by some process with access to the application applying constraint relaxation. If constraints are relaxed until the call to the application results in success, then it is possible for the system to announce what constraint in the caller's requirements has lead

[1] A major potential source of failure, not specifically considered here, would be where the models used by acoustic-phonetic decoding have been insufficiently trained to cope with speaker variability.

to the impasse. Alternatively, or additionally, a suggestive response may be generated by specifying a solution which became possible after such relaxation, or if a number of solutions exist, stating a non-specific solution and inviting the caller to 'browse' interactively. While this approach seems to tally with human cooperative behaviour, there are a number of difficulties:

1. Constraint relaxation is likely to be nondeterministic. It is important to know which constraints to try relaxing first, especially when a number of parameters are involved.
2. the granularity of steps through which constraints are relaxed may be difficult to determine. Some relaxations (such as relaxing times by minutes) may be too fine-grained to make any difference, while others (such as by hours, days) may be too coarse to win acceptability.
3. related to this is the concern that the minimum relaxation necessary to find a solution is carried out—a solution which threw away all or most of the user's specific requirements would be of little value.

These considerations suggest that considerable search may be needed. Because the application is likely to be a component detached from the remainder of the SUNDIAL system, with its own DBMS and possibly accessed remotely over a modem link, this may lead to a severe loss of efficiency. It is therefore important to have good heuristics governing the order and granularity of constraint relaxation.

### 4.2.7 Linguistic phenomena

For these, we may subdivide the responsibilities of the dialogue manager as follows:

1. interpret correctly cues to dialogic, and discourse function; similarly, ensure that the system's use of such cues in its own utterances are correct;
2. maintain contextual registers in such a way that context-specific interpretation of contextually dependent material, such as indexicals and anaphors, is correct; likewise, ensure that the use of context-specific linguistic devices in the system's utterances is consistent with its current representation of context, and update its contextual registers accordingly.

One such contextual register is the referential (or belief) model, discussed in section **??** below. Another is the linguistic history, containing a record of the surface structures of past caller utterances that have been successfully interpretd, and of past system utterances, which enables the resolution of *surface anaphora*. It may also be used to capitalise on the 'input-output principle' [**?**], whereby dialogue participants tend to converge in their use of language, and for the detection of echoes. This is done during parsing, by referring to the linguistic history as well as the static rules of the lexicon-grammar, and likewise during generation.

Linguistic devices that indicate dialogic function are often under-specific; such interpretation may therefore have to take place in conjunction with deep semantic interpretation, and considering dialogue history and expectations. In its utterances, the system may be capable of expressing more nuances than it can detect in the caller's utterances. For example, it may indicate with devices such as increased explicitness and an intenser prosodic contour, that a question is being asked for the second, or third time.

### 4.2.8 Information structure

Replicating the linguistic devices used by humans to structure information economically is an important requirement for any system with natural language processing facilities. The use of a referential (or belief) model enables us to track the evolving focal status of discourse objects. We may appeal here to the notions of global and local focus, advocated by Grosz and Sidner. The belief model contains conceptual objects, which may have arisen because of mention or by default inference. An dynamic ordering on the contents of the belief model is used to track the focal status of objects, enabling:

1. the utterance interpretation process to resolve pronouns and anaphoric definite descriptions;
2. the message output subsystem to likewise be able to use shortened descriptions.

Interpretation of utterances with ellipsis, or non-sentential utterances, depends on these being 'anchored' in the belief model. Again focal considerations, especially those motivated by the states of the dialogue and the task, may be used here.

Topic changes which the system can deal with will be related to change between tasks and subtasks. System-initiated shifts will be regular, and depend on prior (sub)task completion. Caller-initiated shifts are more difficult to handle, especially when only inexplicit cues are given.

### 4.2.9 Prosodic phenomena

Speech recognition technology is as yet insufficiently advanced to enable the recognition of prosodic patterns which are dialogically or informationally oriented. However the generation of prosodic patterns is technically possible, and represents an important resource in voice output—for example, a prosodic contour may be used to indicate that there are more ‘chunks’ to come, or may indicate that a confirmation utterance requires a response: cues that in their absence would probably require unnecessary circumlocutions. Prosodic phenomena are probably somewhat better documented for English than for French, German, or Italian, and it is for English that research into the use of prosody in voice output is largely being applied:

- the global prosodic features on an utterance or chunk, as determined by the message planner, may take into account the type of dialogue act, and the current state of the dialogue module;
- contrastive and thematic features may be applied locally to messages, by evaluating the belief and focal properties of the discourse objects that they contain;
- if discourse topic organisation is parasitic on task organisation, as suggested by Grosz and Sidner, then the voice output subsystem may be used to indicate topic beginnings and endings.

The last case is of course only applicable, when the system currently has the initiatory role and is therefore capable of taking the lead in opening and closing topics. A more general issue is at stake here—because of the one-sided nature of the technology, we are not currently able to take into account the potentially complex prosodic interplay between speakers.

## 4.3 Design Objectives

### 4.3.1 The need for declarative representations

There are a number of reasons why declarative representations are particularly necessary:

1. need for portability across languages;
2. potential need for portability across application domains;

3. by increasing the level of abstraction, it is possible to experiment with various dialogue strategies.

As a minimum, we should require that the dialogue manager is generic, and has no built-in knowledge of the data from linguistic/conceptual/application knowledge bases on which it works. To further increase portability, we may in addition require that domain specific and domain independent knowledge should be separable. A further, independently motivated requirement, is that the representations should be capable of formal description. This would increase our understanding, and allow formal properties to be demonstrated off-line.

### 4.3.2 Separating management of various kinds of knowledge

- different kinds of knowledge may require different styles of management—for example, focus rules will apply to referential knowledge, but not necessarily to dialogue or task knowledge. Sections **??** and **??** discuss these issues in more detail;

- communicative goals with different origins may compete to be realised as a message. For example, task-driven goals require a speedy completion of the task, which may conflict with goals concerned with the reliability of communication. Consider the following dialogue fragment:

  (5) U: i'd like to go to rome next monday morning
  S: there is a flight which takes off at 10 a m is that ok
  U: yes
  S: you will receive your ticket to bonn; bye
  U: urghh WHAT did you SAY?

  This misleading situation is due to speech recognition limitations which can confuse *Bonn* with *Rome*. The failure of communication arises from failing to take account of this fact. We therefore require that the system should have as a persistent goal the validation of caller inputs, as well as the speedy completion of tasks. This may be done either by giving the module responsible for the task responsibility also for validation of task parameters; alternatively, a monitor process may require validation of all inputs.

- The results of local management decisions need to be accessible to other modules. For example, decisions about the probable referential identity of a domain object, and its current focal status, need to be accessible both to modules responsible for linguistic interpretation, and to those responsible for planning/producing output.

### 4.3.3 Respecting users' goals: the need for cooperativeness and a user model

A number of arguments can be advanced for the inclusion of a user model in the dialogue architecture [?]:

- for cooperative dialogue, the system may need to take into account the user's goals and plans. In the SUNDIAL dialogue system such reasoning will be strictly limited: the users goals will be drawn from the set of permissible tasks, and the user's utterances may be linked inferentially to these. Constraint relaxation (obtaining a match between the user's wants and database availability) will be based on heuristics—there will be no attempt to reason about underlying motivations.
- The system may make assumptions about the user's beliefs. These are treated as defaults, for which confirmation, implicit or explicit, is required. More sophisticated user-modelling classifies users, for example, according to their degree of expertise. We make no attempt to do this here;
- the system may wish to check what effects a planned dialogue contribution will have on the user. In planning a message, for example, the system may build a 'scratch user model', in order to reason about the effects of various choices in building a referential expression.

In addition, certain aspects of keeping a discourse history may be viewed as part of keeping a model of the user:

- maintaining a history of those referential objects in the domain introduced by the user;
- maintaining a history of linguistic expressions employed by the user.

However, these discourse-oriented aspects of user-modelling are perhaps more usefully viewed as part of the larger task of maintaining referential and linguistic histories for both speakers, system and user—see sections **??**, **??**.

## 4.4 Approaches to knowledge representation

Issues of declarativeness, and separation of concerns, presuppose that structuring devices are available for the representation and efficient propagation of knowledge, and that these are as generic as possible. In this chapter, we review the knowledge representation issue, from a number of viewpoints. We begin by distinguishing between *static* and *dynamic* knowledge; then discuss what knowledge should be contained in a knowledge base representing domain objects. We discuss approaches to handling inference, and finally present a series of functional requirements for a knowledge representation language for SUNDIAL.

### 4.4.1 Static and dynamic knowledge bases

We may distinguish between static knowledge bases, or *knowledge definitions*, which do not change within a session, and dynamic knowledge bases, evolving representations of the world, which do. In line with common knowledge representation practice, dynamic knowledge may be treated as patterns of instantiations of static knowledge. There are two ways in which dynamic and static knowledge bases may interact:

1. dynamic knowledge may be created according to static knowledge templates. This may be preferred if there is no need to access class information, once individuals have been created;
2. dynamic knowledge may be linked directly to its static counterpart. There is scope for structure sharing here, and advantages to be gained if classes, as well as individuals, need to gain focal importance from time to time.

The above discussion assumes that the definition of classes is frozen, and hence static knowledge; this need not be the case. It is wrong also to confuse level of specificity with the static/dynamic distinction.

In the following discussion, we shall use the term *knowledge base* to refer to dynamic knowledge bases, and reserve the term *knowledge definition*, for the case of static knowledge bases.

### 4.4.2 The nature of information in a referential KB

#### 4.4.2.1 Temporal requirements

There are a number of alternative approaches to representing and handling information with time dependencies:

1. a naive database approach, which keeps temporal indices for every fact it stores;

2. the situation calculus, which indexes facts to situations, and represents events as transitions between situations;
3. some form of temporal logic.

A major drawback of the first approach is its inability to deal with generalities and partial information. The second suffers from a syndrome known as the 'frame problem': how to economically represent the persistence of facts which are unaffected by events. Temporal logics do not necessarily suffer from these disadvantages.

A number of choices exist, regarding the use of a temporal calculus:

1. the representation may be interval or point based;
2. a modal temporal logic may be used, or relations holding over time may be 'reified' hence allowing the use of a first-order logic;
3. temporal relations may be confined to those of precedence, overlap and subsumption; alternatively causal relations may additionally be defined.

A reified interval-based approach is that of Allen and Hayes [**?**]. The temporal calculus of Kowalski and Sergot [**?**] may be viewed as a an extension of this which makes assumptions about causality. Because of the reification, and because temporal knowledge is generally partial, there are inferential overheads in the use of a temporal logic.

#### 4.4.2.2 Ontological requirements

It is important that an ontology of objects is defined, to ensure portability and generality, and because constraints in the resulting sorted logic are more easily stated. We may for example divide objects in the representation into those which have associated time periods, and those which are persistent. Examples of the first could be:

a departure-event
a flight-event
the event of making a reservation
the event of an individual checking-in
the period during which a passenger waits for his flight to take off
the period during which the aircraft is airborne, etc

Examples of the second would be places, carriers, hotels, planes, individuals.

Additionally both sorts of object may be singular or collective. A collection of objects must have more than one member, all of which comply with the defining properties of the collection.

Within this basic ontology, representational efficiency (and portability) can be enhanced by adopting some taxonomic organisation. This also is needed to reflect the ability of natural language to refer with to objects with generic or specific descriptions. The resulting *sortal lattice* may be viewed either from the point of view of a class hierarchy in the static knowledge base, or as a description of dynamic objects with differing levels of specificity.

### 4.4.3 Representing dialogic knowledge

A number of approaches to describing dialogue organisation are available.

1. An unstructured approach respects only local organisation. Turns can be initiatory, or dependent on earlier turns, either as continuations or as completions. Otherwise there are no constraints.

2. An approach based on intentional structure attempts to explicitly link in every contribution, as operators in the evolving intentional structures of speakers;

3. An 'autosegmental' approach attempts to describe the merging/overlapping of dialogue acts and turns, using for example a temporal calculus representation;

4. A 'dialogue grammar' approach presents rules for well-formed dialogues. For example, in the work of Roulet and Moeschler [?] we have the following pattern of constituency:

| category | level | constituents |
|---|---|---|
| transaction | task, subtask | exchange |
| exchange | adjacency pair | intervention |
| intervention | dialogue act, turn | |

   It may be possible to define a tangled hierarchy, for example with exchanges having exchanges as constituents.

These approaches are more or less equivalent, insofar that they make normative predictions about dialogue structure. They seem to differ to the extent to which structural phenomena are placed directly in the dialogue rules, and the extent to which they are an emergent effect of some underlying structure such as intentional structure. They differ also in their descriptive power—the autosegmental one being the most descriptive; and in their predictive power—the dialogue grammar being the most predictive, albeit sacrificing a certain amount of the flexibility observed in human-human dialogues.

So far as the long term research goal of modelling human dialogues is concerned, the autosegmental approach seems the most promising; given the current need for constraining power, the plan-based or dialogue-grammar based approaches appear to offer more immediate technology progress.

### 4.4.4 Addressing KBs

Brachman *et al* [**?**] present a rigorous specification of the interface between a Knowledge Base and an outside client. They do this by treating a knowledge base as an abstract datatype, and specifying a series of permitted operations on this, including: *NEWKB, ASK, TELL, SUBSUMES, DEFINE.* This has the dual advantage of presenting modular interfaces to KBs, so that different implementations can be experimented with, and permitting reasoning to be done about a KB at a formal level. A number of additional operators might be required for the SUNDIAL Knowledge Representation, for example;

| Operation | gloss |
|---|---|
| UNIQUE? | returns a truth value indicating whether a description represents a unique object |
| EXISTS? | in the case of a description being satisfied by an object in the KB, an index is returned; otherwise, $FALSE$ |

Insofar that the dialogue architecture may be viewed as a number of cooperating knowledge bases, the messages which pass between these may be structured as query or update operations of this kind.

### 4.4.5 Inferential requirements

#### 4.4.5.1 Levels of complexity

The analogy is often made between a referential, or domain KB, and a database, or logical theory:

- at a simple level, a collection of facts, expressed in some form of logic or as simple semantic nets or in the form of a database, may be used to represent the world. Such a representation, though straightforward to implement (eg, in Prolog), suffers from the limitations of the closed world assumption, and from the absence of constraints;
- representations which fall within the expressive power of first order predicate logic allow in addition to facts and rules, arbitrary constraints which are capable of leading to inconsistent states of affairs. There are implications for the kind of inference engine used. Unlike a database representation, which can rely entirely on backward chaining, some forward chaining is required, to ensure that inconsistent states of affairs do not come about. Introducing temporal constraints also may have repercussions for the kinds of reasoning that can be used. Although Kowalski and Sergot describe their event calculus in terms of rules which are capable of being implemented using Prolog-style backward chaining, the more comprehensive rules of Allen [?] require a more general constraint-satisfaction method;
- defaults may be introduced. As a result we no longer have a simple classical monotonic logic: inferences based on an earlier default assumption may need to be withdrawn if an inconsistency later arises. Handling defaults is discussed in more detail in ??.

It is assumed that we shall need something approaching the expressive power of first order predicate logic, together with the capability to handle defaults.

#### 4.4.5.2 Non monotonicity

There are a number of reasons why a Knowledge Base need not be monotonic:

1. Inferences may be built on default assumptions; once such an assumption has been contradicted, it will prove necessary to withdraw any inferences that depended on it;
2. Hypotheses from the linguistic analyser may also lead to inferences which later become invalidated;
3. The user may wish to explore various hypothetical states of affairs, without committing to one of them.

As pointed out by Kowalski and Sergot [?], it is possible to encompass seeming nonmonotonicity within a temporal reasoning scheme, by bounding periods associated with the system's belief in defeasible facts by events at which that belief ceased to hold. This however is carrying temporal reasoning into the meta-, self-referential domain, which seems undesirable. It also fails to deal with implementational issues of belief *maintenance*.

A reason maintenance system works in tandem with a problem solver, recording its findings in a network of justifications. There are several benefits to be gained from this:

- The results of inferences are kept in a cache. If indexing is efficient, it may be less computationally expensive to retrieve such a result, rather than having to derive it again.
- If inconsistencies are detected, it is possible to follow chains of justifications back to find and remove the culprits. This is in contrast to chronological backtracking, where good work may need to be undone before the reasons for failure are detected.

Reason maintenance systems are generally divided into two main types: justification-based (JTMS) and assumption-based (ATMS). The former are capable of representing only a single consistent state of affairs, while the latter may represent several alternative consistent states of affairs, or 'contexts'. There is a trade-off between the usefulness of the one and the other: an ATMS is likely to be useful when several sets of alternative assumptions need to be explored; a JTMS will be more efficient in cases where complex reasoning within a single set of assumptions is called for—with an ATMS there would be the unnecessary overhead of maintaining more than one set of assumptions.

The usefulness of reason maintenance systems also needs to be weighed against i) the 'encoding problem', of determining which inference results are worth storing, which facts are to be treated as assumptions, etc, and ii) the overhead involved, in space and accessing and updating times, in maintaining large networks of justifications. A partial solution to these problems would be to maintain dependency information only for those nodes which wholly or partially depended on default assumptions, or on uncertain hypotheses. Additionally, defeated assumptions could be eventually removed by a garbage collection mechanism. Such techniques would not however be wholly suitable for the requirements of maintaining hypothetical alternatives [**?**].

Whether or not a reason maintenance system is needed, will depend on the extent to which non-retractable inferences (for example, those which lead to system utterances) are permitted to depend on defeasible knowledge. An extremely conservative strategy, for example, might interactively ask confirmation for each assumption before incorporating it.

#### 4.4.5.3 The relevance of logic

The above discussion tends to presuppose that logic is the ideal representation. Such an argument has been made in the past; but as Woods [?] points out:

> ...one can axiomatise any rule-governed behaviour that one can rigorously specify, whether its nature is deductively valid or not, logic-like or not. But the difficult issues mostly stem from determining what the behaviour should be and how to bring it about. This work remains whether the medium is predicate calculus or some other representational formalism.

Woods finds predicate logic wanting, both in expressive adequacy (the lack of straightforwards means for representing persistent objects, inheritance, partial information) and in notational efficacy (the lack of structure sharing, and the need for additional indexing mechanisms).

### 4.4.6 The SUNDIAL knowledge representation language

The representation must be capable of capturing all the information necessary to interpret, generate and reason about natural language within information service dialogues. Some of these requirements are listed below

- **Partiality** Many processes, such as truth and inference, are based upon partial, rather than total, knowledge of entities and relationships. Of course, the notion of partial information can only be ground out if the corresponding notion of total information is also ground out.
- **Formalization Of Content** Many processes are dependent upon the content of entities and relations, rather than just their 'formal' properties as traditionality conceived. This militates against the treatment of entities and relations as unstructured atomic representations.
- **Granularity** Processes can vary in the level of specificity to which they are attuned: ie. some processes operate on the basis of fine-grained distinctions whereas others operate on the basis of coarse-grained distinctions. The representation must be capable of supporting both as well as specifying the relationship between them.
- **Entities and Relations** The representation must be capable of capturing the similarities and differences between entities and relations.

- **Complex Objects** The representation must be capable of describing complex entities and relations. For example, it must be possible to represent complex entities such as mass entities and set entities.

- **Types and Individuals** The representation must support the representation of both general types of entities and relations as well as particular individual of that type.

- **Property Constraints** The representation must supported the representation of constraints between different properties of the same entity or relation.

- **Contextualization** Some processes are contextualized in the sense that the process only occurs within a certain situation. Of course, distinction between the process itself and the conditions under which it applies should be maintained. Likewise, properties of entities only holds within certain situations. For example, a *carrot* can be an individual entity in one situation, but a mass entity in another (ie. when it has been grated!).

- **Dynamics** The representation must facilitate representation of changes in the information state so that, for example, we can come to know more about an entity such as the flight which the user is enquiring about.

- **Inference and Defaults** The representation must be capable of supporting inference and default reasoning. Inference is taken to include both monotonic and non-monotonic reasoning.

## 4.5 Histories

### 4.5.1 Maintaining a history of domain knowledge

There are two potentially conflicting requirements:

1. Because knowledge about the domain may be built up during the course of the dialogue by incrementally adding to partial descriptions, we need to be able to update the descriptions of objects. The resulting belief set at any time should be capable of representing (part of) the world, independently of the order in which such knowledge was arrived at;

2. Because the addition of new knowledge may be nonmonotonic, we need to have the facility for backtracking and undoing the defeasible parts of object descriptions.

One means of maintaining a single consistent set of objects, whilst keeping open the possibility of analysing the source of knowledge, is to represent objects as versions, which share an object identifier, but are chained, each object being derivable from a predecessor, via a set of changes: additions, deletions, substitutions. The current version of an object is therefore the most recent in the chain, or the most recent in the current perspective. In the case of long chains, some amount of compaction of their earlier portions may be necessary.

Reliability checking may be implemented by following back these object histories, and checking unreliable portions. Revisions are not destructive of old information, so that at a certain level of description the representation is monotonic. Unlike a complete TMS or ATMs, we are not concerned with re-deriving results which may have to be undone because of retractions.

### 4.5.2 The dialogue history

In contrast to the referential KB, a dialogue history needs to represent:

1. the nature of the labels on individual acts;
2. their relative sequencing;
3. their relations (parent, sister) with one another.

Where dialogue acts refer to domain objects, there exists a design choice:

1. copy those domain objects, in their original form, into the dialogue history;
2. simply provide pointers into the referential KB.

The second alternative is more economical, but there is lacking direct knowledge about how much was originally said about an object, and what objects were in focus at the time. It is not clear however that humans are capable of doing this, further than a few turns back. See **??** below.

#### 4.5.2.1 Respecting dialogue coherence: the need for a proper dialogue history

The main purpose of the dialogue history is to provide references to what occurred during the dialogue. Thus, it keeps tracks of the various turns and allows to know *who and what and in which context* an intervention has been uttered. When representing an intervention in the dialogue history we have to provide with the owner of it, its purpose and the context. What is certainly ambiguous is the term “context”. With context we mean what are the current knowledge of each dialoguee about both the task and the dialogue (and probably personal feeling about the other). As we can see, these knowledge belong to the belief module, the dialogue module and possibly to the task module.

The simplest way of representing a dialogue history is to have one frame for each move and the whole history is an ordered list of frames [**?**]: each frame has the following entries: the number of the intervention (which helps for chaining the turns), the owner, the theme (a pointer to a theme lattice denoting synonyms between objects), a pointer to the user knowledge when the turn occurred and lastly a pointer to the system reasoning history (tracks).

It may not be valuable to have a non structured representation of the dialogue history. One crucial factor for maintaining cohesion and coherence of the dialogue is to keep track of the current topic of the conversation, thus having only the theme of sentences seems to be not enough. In fact, the dialogue history should reflect also the structure of the dialogue, that is how exchanges are organised.

In the Esteam project, the dialogue history is completely tied to the task management history. As the internal representation of the task is a tree, the dialogue history captures the same organization: when we are talking about a subject then we are somewhere in the task tree and we tag the current node as “visited” (more precisely, they state whether the discussion about the item is completed or not). Thus, changing the current focus during the dialogue is synonymous to move through (back or forward) the task tree. This approach is open to criticism for two main reasons: first it fails keeping distinguished the task from the dialogue management which is one of our main objective, second whenever one dialoguee comes back to an already discussed topic, the new information is written, thus this forgets the previous information and then this partially describes the history.

A more general theory is the one proposed by Grosz and Sidner [**?**, **?**], where the dialogue history could be viewed as a structure of discourse segments. Basically, a discourse segment serves one intention (the intentional state), they are organised via the dominance and precedence relations. In parallel, there is a stack of focus spaces: each discourse segment points to a focus space, the current focus space is then the active discourse segment. A focus space corresponds to the “attentional state” which is a collection of objects discussed during the dialogue segment.

This intuitive approach seems to be very efficient as far as we don’t have to face with topic resumptions and topic dropping. A pure stack mechanism (and then a Push–Pop paradigm) fails in keeping a coherent stack when exchange endings are not explicit : a “pop” will miss! Thus, special ad hoc processes are suggested which in fact degrade the stack mechanisms, as described in [?].

Another problem which occurs when a stack is used, is that we have only a “short term memory” of what happened during the dialogue. When an exchange is closed, you simply pop it and then forget it for the rest of the dialogue. However, maybe especially in oral communication, backtracks are always possible, such as “what did you say ?”. So again, you must provide adjuncts to the stack paradigm. It remains unclear how long we have to keep “items” in memory. What can be said is that it is not because a new discourse segment is open that we could forget the preceding one (if it don’t dominate the new one). However, we copula say that after a mutual agreement of the locutors on the segment closing then we can loose its history (but not its effects) but the notion of “mutual agreement” is clearly obscure!

One possibility is to keep track of the dialogue is to memorize it structurally, as in Grosz’s segmentation proposal but keeping all in mind as in the LOQUI project [?], [?]. The problem is to clearly state what should be in. In LOQUI, the dialogue history is the track of the dialogue grammar which manipulates dialogue acts. The history is represented as a tree where leaves are the moves (or turn) stamped with the corresponding dialogue act. The upper level is the level of exchanges and the upper one, discourse level. What is missing in their history is the Grosz’s stack of focus spaces; it is important to know when (where in the history) objects have been updated and in which circumstances. Thus, a possible proposal for our dialogue history is to mix up LOKI proposal with Grosz and Sidner’s proposals. The dialogue history could reflect the track of the dialogue grammar (or model) with more information attached to each node to specify the focus space, and the intention.

### 4.5.3 Linguistic histories

It may be desirable to maintain linguistic representations across turns, for the following reasons:

- a repair may make sense only at the linguistic level;
- likewise for substitutions or extensions to a previous utterance;
- when generation and linguistic processing have access to a common linguistic history, it may be possible to organise dialogues in such a way that convergence of descriptions takes place,

It would be uneconomical to store an entire chart, but that portoin of a chart leading to a spanning representation (with packed or unpacked semantics) may be kept. Such a set of *surface linguistic representations* may be indexed into by the following means:

- in the surface semantic representation, which has yet to be anchored, variables stand as placeholders for object indices;
- when referential interpretation has taken place, these variables have indices in the belief module assigned to them;
- it is subsequently possible to locate linguistic expressions whose semantics contains a given index. One may wish to locate, for example:
  - the first such expression;
  - the most recent such expression;
  - the smallest such expression, etc

### 4.5.4 Focus and indexical knowledge

These are not so much histories, as pointers to areas needing highlighting, within independently motivated history structures. They are included here because their requirements may have repercussions for the representation of the histories themselves:

**Attentional focus** or *discourse topic* [?] marks out topical areas in a layered fashion, resulting in descriptions which obey scope rules akin to those used for identifiers in block-structured programming languages. This focussing structure is superimposed on referential material; however the motivation for changing focus of attention has its source in the dialogue structure;

**Local focus** This more local focussing mechanism is principally used to maintain focal persistence between successive turns, and may be used to predict local anaphoric coherence (deaccenting, pronominalisation). Because of the locality requirement, it may be implemented on the interface between linguistic histories and the referential KB. A possible algorithm for local focus tracking is given in [?];

**Frames of reference** include markers corresponding to the deictic "now" and "here". In order to describe the tenses of natural language, it is common to have a number of temporal frames of reference [?]:

1. event time;
2. reference time from which event is viewed;
3. time of speaking.

### 4.5.5 General issues concerning the maintenance of histories

A number of issues require further exploration. Regarding the permanence of historical information, we assume that this is not saved from one session to the next. We have seen that sequencing information may be more or less important in structuring histories. The question still remains whether the system should be allowed to 'forget' after a time, freeing up available memory; psycholinguistic evidence suggests that human verbatim recall of spoken material is short-lived, in contrast with the recall of gist.

We have indicated the advantages of a degree of structure sharing, for example between the referential KB and the dialogue history. A radical approach would additionally use homogenous representations, so that the same language was used to talk about the terms of either history. One result would be that dialogue acts could be considered on a par with other objects within the referential KB. There are a number of objections to such a proposal:

1. the issue of embedded attitudes is raised. It is not clear what the proper inferential treatment needed would be;
2. we would have the anomalous situation of speakers being able to refer to arbitrary dialogue acts in the dialogue history as objects of discourse; not only this, but it would be possible to refer to exchange and topic structure also.
3. It is probably not the case that the complexity of temporal representation suggested above for domain objects, is needed for dialogue objects; any structures which preserve ordering information will suffice.

On the other hand, a limited amount of self-reference can probably be implemented in terms of the dialogue architecture, by allowing statements describing for example the contents of, and updates to, the various knowledge bases.

## 4.6 Conclusion

In this chapter, we have reviewed the various design principles and technical approaches which could be adopted for the implementation of a Dialogue Manager. The review has shown that in some cases, it is not clear from existing knowledge and experience which of several alternative appraoches would be best. Nevertheless, we need to make decisions, for the moment perhaps arbitrary ones, between these alternatives in order to make progress towards the first implementation of the Dialogue Manager. Later evaluation work and our own experience will help to determine whether these decisions were correct and may suggest that other possibilities should also be carefully explored.

Furthermore, it would be unrealistic to plan a Dialogue Manager which would cover *all* the phenomena outlined in Chapter 3 and which would also tackle *all* the issues mentioned in this chapter. Hence, decisions have to be made about which functions should be supported by the demonstrator and which should be left for later versions. The next chapter begins with a section on 'Priorities' which is based on the results of our deliberations about which functions should be included in the intermediate demonstrator Dialogue Manager. The rest of the chapter provides a specification of this limited Dialogue Manager and describes its internal architecture. Later versions of the demonstrator will enhance its functionality, with the aim of ultimately dealing with all the phenomena identified in Chapter 3.

# Chapter 5

# Functionality and Architecture

In previous chapters, the requirements for a dialogue manager in the SUNDIAL system have been discussed and general design principles and approaches have been considered. In this chapter and the following one, we move to specifying the functionality of the intermediate demonstrator Dialogue Manager. Not all the phenomena identified earlier can be expected to be handled by this intermediate system, so it necessary to select those which will, at a minimum, provide a working system that can be evaluated. This chapter therefore begins by listing a set of phenomena in terms of a ranked list of priorities for implementation. The intention is that the functionality of the intermediate system will allow it to deal with all those phenomena noted as 'essential', and as many of the others listed as possible.

Having defined the minimum and expected coverage and capabilities of the system, the chapter then considers how the desired functionality may be provided. An overall, abstract architecture for the Dialogue Manager which partitions it into a number of 'sub-modules' is described. Each sub-module is considered in turn. In the course of the specification of these sub-modules, particular approaches to providing functionality are advocated, selected from those reviewed in the previous chapter. While some details are left open for experimentation during the course of implementation (and these are identified as such in the course of the chapter), most of decisions have already been made and are noted here in this chapter. These decisions have been informed by prototyping work carried out by partners while this specification was being prepared and written.

## 5.1 Priorities

Chapter 3 includes a very extensive list of the phenomena which can be observed in human–human and human–computer dialogue. It would be unrealistic to expect the intermediate demonstrator to handle any but a small subset of all these phenomena. In this section, we set out a list of those phenomena which we aim to be able to deal with, ranked from essential to desirable but unlikely to be achieved.

In constructing this list, two major criteria were used. First, phenomena which occur frequently and which have a significant rôle in human-computer dialogue were favoured over phenomena which are incidental, or found mainly in human-human dialogue. Second, phenomena which could be handled relatively easily, or where the techniques required were already well known, were favoured over phenomena where it was unclear which approach would be best or where methods are experimental or unknown.

Tow items were regarded as essential. The functionality of the SUNDIAL system as a whole demands that the Dialogue Manager be able to send the Linguistic Processor predictions about expected dialogue acts. Even though constructing such predictions is likely to be challenging, this item was included as 'essential'. Secondly, it was considered an essential characteristic of a dialogue manager that the dialogue that it generates be adaptive (to the ongoing interaction and, in particular, to the recognition achieved so far) and thus this feature was also listed as essential.

In the table below, 'Essential' means commitment to achieving an implementation task; bands 'A' to 'D' reflect different levels of priority, 'A' being the highest.

| Priority level | achievement |
|---|---|
| Essential | send predictions to parser<br>adaptive dialogue |
| A | intersentential anaphora—interpretation<br>referring expressions—generation<br>more than one task |
| B | ellipsis in input<br>referring expressions—interpretation<br>summarisation<br>chunking |
| C | intersentential anaphora—generation<br>ellipsis—generation<br>overlaps<br>interruptions<br>prosody |
| D | complex quantification, negation<br>complex task inferences<br>database updates<br>extensions to utterances |

A minimal system will be the simplest one so defined, whereas a maximal one will have all of the listed functionalities implemented. The actual implementation will lie somewhere between these limits.

## 5.2 Overview of the architecture

In Chapter 1, the Dialogue Manager was introduced as a single software module which co-operates with other modules in the Sundial system in order to enable an interaction between the system and the user, rather than a simple exchange between them. The overall task of the Dialogue Manager can be divided into several subtasks, and for the sake of modularity each of these subtasks can be assigned to a functional sub-module.

In the following sections, these tasks and their interrelations are described in terms of such modules. It has to be kept in mind, however, that the division of the Dialogue Manager into sub-modules is an analytical one and will not necessarily be reflected in the implementation of the Dialogue Manager.

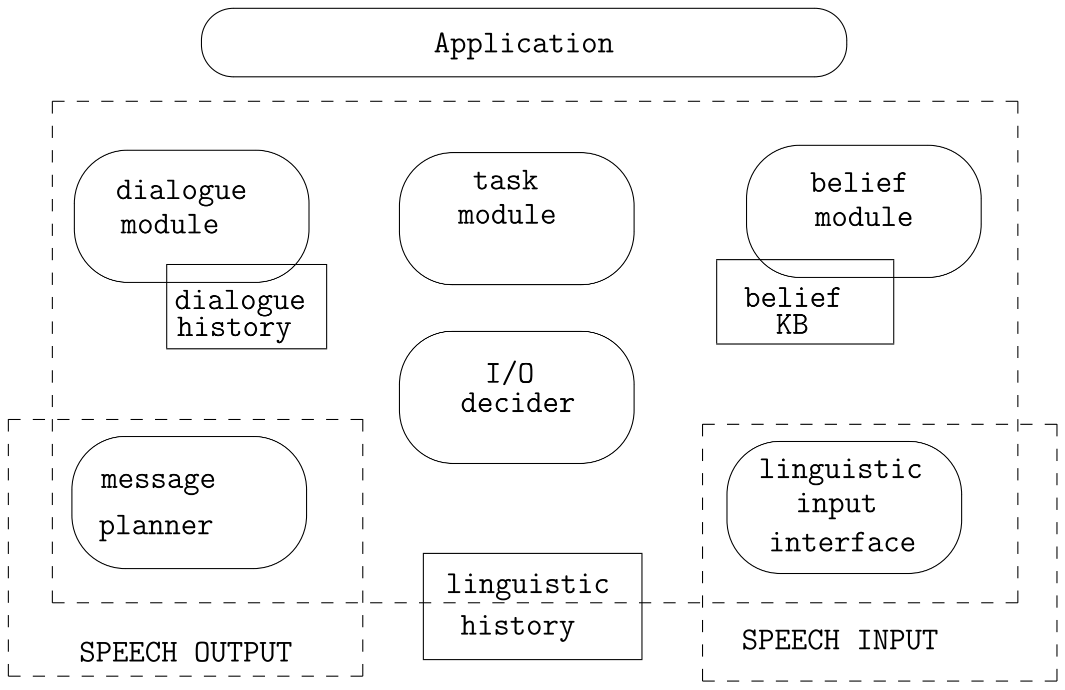

Figure 5.1: Schematic architecture for the Dialogue Manager

### 5.2.1 Decomposition into sub-modules

Figure **??** shows a schematic view of the architecture, indicating the sub-modules. The major features of this architecture are:

- The architecture is concurrent; the modules are to be viewed as communicating experts;
- The various modules are responsible for maintaining their own histories (or *knowledge bases*), and channelling lookup/update requests from other modules; knowledge is therefore *distributed* among components—for example, knowledge resulting from the interpretation of an utterance may be distributed amongst the belief module, the dialogue module, and the task module;
- The Input/Output Decider has the task of arbitrating between competing communicative goals arising from other modules, and thus is responsible for deciding what to say next.

In the implementation, the functionality of the Input/Output Decider will be distributed amongst the dialogue module, the message planner, the task module and the belief module, but it has been shown separately in this figure. For clarity of presentation in this document, discussion of the functions of the I/O Decider will be merged with those of the dialogue module.

### 5.2.2 Interactions between modules

First, let us briefly present how the dialogue manager works as a whole. At each turn in the dialogue, the dialogue module constructs dialogue goals on the basis of possible dialogue structures. Depending on whose turn it is to speak, these dialogue goals provide either dialogic descriptions of the next system utterance or dialogic predictions about the next user utterance. When it is the system's turn, goals from the task module, such as requests for missing task parameters, goals from the linguistic interface module such as requests for the repetition of missing words, and goals from the belief module arising, for example, from referential failure, are ordered and merged with the dialogue goals. The resulting goals are then sent to the message planning module for planning and eventual realisation by the Message Generator.

When it is the user's turn to talk, task and belief goals are ordered and merged with the dialogue goal to form a prediction. This is sent, via the linguistic interface module, to the Linguistic Processor. When the user speaks, a representation of the user's utterance is passed from the Linguistic Processor to the linguistic interface module and then on to the belief module. The belief module assigns it a context-dependent referential interpretation suitable for the task module to make a task interpretation and dialogue module to make a dialogic interpretation. This results in the construction of new dialogue goals. The cycle is then repeated, to generate the next system turn.

This is necessarily a very much simplified overview of the processing which takes place inside the Dialogue Manager. The following sections of this and the next chapter describe each sub-module in more detail and indicate its rôle in the process as a whole.

The architecture respects the design goals stated in chapter 4 (sections 3.1 and 3.2): task management has been separated from dialogue management. The architecture is common to each of the application domains and to each of the demonstrator systems.

Now let us describe each three main modules: the task module, dialogue module and the belief module. SIL, the basis for the knowledge representation formalism to be used in the dialogue manager, and the message planner module and the linguistic interface, both of which are primarily concerned with interactions with other parts of the Sundial system, are all described in Chapter 6.

## 5.3 The task module

### 5.3.1 Functionality

The task module is responsible for task management. Its functions are to establish the various parameters needed to access the database and to obtain solutions to the user's requests. Requests for parameters and the solutions sent to the user need to be formulated according to cooperative principles.

The module needs to handle the following functionalities each of which are described in detail in the following sub-sections:

1. establish task and task parameters,
2. check specificity,
3. check reliability,
4. access the application database,
5. present responses from the database,
6. relax constraints on the user's request,
7. make task specific inferences.
8. supply default knowledge

#### 5.3.1.1 Establishing Tasks and Task Parameters

Enquiries may concern any object that the system has knowledge about. Enquiries about some property of an object are best established by building up a description of that object. The task module checks what information about particular objects, if any, is available in the belief module. If more information is required, system dialogue acts are instigated by issuing appropriate communicative goals for prioritising.

The task module could receive precise or underspecified information about one parameter or even no response at all. In each case this must be dealt with as a task parameter answer. If the response is not sufficiently precise to enable a solution to be obtained from the application database, a negotiation with the caller has to be initiated. The structure of this negotiation is a matter for the dialogue module. Thus, the number of exchanges may vary from one caller to another to establish the same parameter.

The task parameters needed from the caller will vary according to the application domain and according to the task (e.g. make a flight reservation, enquire about a particular train, list all the hotels with vacant rooms in a town). However, the basic mechanism of collecting sufficient parameters to a sufficient degree of precision to allow a database access, and then returning an answer, is common to all these applications. It is this basic mechanism which is provided by the task module.

As an example, the task parameters which would be required for the flight reservation task are:

- Departure city
- Departure airport
- Arrival city
- Arrival airport
- Departure date
- Departure time
- Arrival date
- Arrival time
- Flight number
- Class
- Company
- Price
- Number of seats
- Passenger name(s)
- Phone number of passenger(s)

#### 5.3.1.2 Checking specificity

Checking specificity involves determining whether there are enough parameters known for the enquiry object to be well defined. Specificity constraints need to take into account:

- the order of establishing task parameters;

- the set of defining parameters for a task may include alternatives; for example, instead of several parameters indicating places and times, a flight number (if known) is sufficient to identify a flight.

The order in which parameters are to be established is determined by the the nature of the task and domain-specific functional dependencies between parameters. The ordering can be obtained from an analysis of each domain and from observation of the preferred ordering used by callers in the simulations (see Chapter 3).

In addition to constraints from the user and from functional dependencies, there are instances when the question "are enough parameters known for the object to be well defined?" can be answered in collaboration with the database. For example, there is no need to know the departure time of a flight if there is only one flight a day to a destination. A procedure for establishing whether sufficient parameters have been obtained to answer a request may thus involve multiple database accesses. If a database access is slow, this could cause intolerable delays to the interaction. Thus, determining the minimal set of parameters through database access will be kept as an option, available if the application permits rapid access.

#### 5.3.1.3 Checking on the reliability of task parameters

Two types of check need to be made on task parameters before a solution to the user's request is made to the application database.

- The first type of checking is pragmatic, testing the existence or coherence of a parameter (e.g. the specified flight number exists in the timetable, date information is coherent and that the return date is after the date of the outward journey ). Whenever an inconsistency is detected, the task module requests the appropriate system output behaviour.

- The value of some parameters will be inferred by default by the task or belief modules. In these cases there is a doubt about the parameters' reliability. These inferences concern only parameters which have not been supplied by the user. The fact that these parameters are inferred will be made known to the belief module, because it may be that the caller will subsequently reject the inferences after the answer has been presented by the system.

#### 5.3.1.4 Accessing the application database

To access the database, the task module has to have knowledge about the data base request language, SQL for example. Thus it should know how to translate the internal representation of objects into a well formed request in this language. In addition, it will receive database answers, probably as tables or relations, and must be able to translate them into an appropriate internal representation for further processing by the dialogue and message planning modules.

#### 5.3.1.5 Presenting responses

Data base answers should not be presented in bulk to the caller. The task module should analyse answers according to the user's request: if there is one response which fits what the caller desires exactly then this could be the only answer. Otherwise, a careful ordering or an summary should be provided. A summary is required when many answers fit the caller's request, otherwise, the answers should be presented in order of interest from the most interesting to the least.

Items such as dates, time, companies should be the basis for a summary. The result of summarisation could lead to messages like "There is one train daily" or "There are two Air France and one British Airways flights".

#### 5.3.1.6 Constraint relaxation

If the caller asks for information which does not exist, a cooperative system will relax the user's constraints and reformulate the question to give a helpful answer. For example, if the user asks about the arrival time of the train to Munich which leaves Ulm after 11pm, when in fact the only late evening train leaves at 10.58, the system should not simply report that there is no train which leaves Ulm after 11pm. Instead, it should relax the constraint that only trains leaving after 11pm are of interest and should provide a (suitably marked) answer about the 10.58 train.

Thus, whenever the data base sends an empty set of solutions, the task module should formulate another data base request by relaxing appropriate constraints. Such a new request is built up according to knowledge which must be domain specific, but the techniques for performing constraint relaxation will be generic and will form part of the task module's functionality.

#### 5.3.1.7 Task specific inferences

As the previous sections implied, the task module needs to have knowledge that will enable it to infer task specific information. Because the Sundial applications are quite simple, we foresee only one kind of inference, based on functional dependencies between parameters. These inferences are to be used to regulate the flow of task goals. The task module will systematically try to infer parameters before setting in motion goals to request them of the caller. Moreover, it must be able to suppress such a goal as soon as it becomes irrelevant. For example, if the task module is waiting for the departure time and obtains the flight number from the caller, it should suppress the former goal, which can be deduced from the flight number. Such an inference depends on knowledge within the task module that there a flight number corresponds with one and only one departure time.

#### 5.3.1.8 Supply default knowledge

The use of default knowledge has been observed in the BA, AF and FS dialogue corpora. They allow the task module to know default values for some parameters without asking the caller for them. As an example, the following default rules have been identified from the French simulation corpus,[1]:

$$\frac{Mdeparture\_city\_one\_way(Paris)}{departure\_city_{one\_way}(Paris)}$$
$$\frac{Mclass(economic)}{class(economic)}$$
$$\frac{Mnumber\_passenger(1)}{number\_passenger(1)}$$
$$\frac{Mnumber\_passenger_{one\_way}(x){:}Mnumber\_passenger_{return}(x)}{number\_passenger_{return}(x)}$$
$$\frac{Mtype\_travel(two\_ways)}{type\_travel(two\_ways)}$$
$$\frac{departure\_city_{one\_way}(x){:}Marrival\_city_{return}(x)}{arrival\_city_{return}(x)}$$
$$\frac{arrival\_city_{one\_way}(x){:}Mdeparture\_city_{return}(x)}{departure\_city_{return}(x)}$$
$$\frac{want(user,company_{one\_way}(x)){:}Mwant(user,company_{return}(x))}{want(user,company_{return}(x))}$$

A *default* is any expression of the form:

$$\frac{\alpha(x){:}M\beta_1(x),...,M\beta_n(x)}{w(x)}$$

where $\alpha(x), \beta_1(x), \ldots, \beta_n(x), w(x)$ are well formed formula, over an alphabet $A$ of variables, whose free variables are among those of $x = x_1, \ldots, x_n$. $\alpha(x)$ is called the *prerequisite* of the default, and $w(x)$ is its consequent. $M$ is to be

[1]The notation used here is taken from [[?]] and may not be the internal representation of default rules used in the SUNDIAL system.

read as “it is consistent to assume”. The consequent of a default has the status of *belief*. Thus, the first default in our example list is to be read as: if it is consistent to assume that the departure city of the one way is Paris, then we are allowed to believe that the departure city of the one way is Paris. The term consistent here is to be intuitively interpreted as: $X$ is consistent unless $\neg X$ is provable.

This list of defaults is not complete, but it gives an idea of the amount of default knowledge the task module should have for the French airline reservation application; the other applications will require similar sets of rules.

### 5.3.2 Task module interactions

This module will send requests to the I/O Decider about the parameters it needs and about the task type it is currently involved with. It will receive parameter values (or indications about these values such as “unknown”). These values may be new instantiations replacing old values if users have changed their mind.

The task module will send status reports on the objects it delivers. For example messages like “no answer + constraint relaxation gave this result” will allow the dialogue module to present the failure first, and then the results obtained through constraint relaxation (see chapter 3 section 3.4.2). More generally speaking, the task module will send “failure apologies” with an accompanying status, depending on whether they occur due to inconsistency, constraint relaxation or blocked inferences. Also results after constraint relaxations will be accompanied by the name(s) of the relaxed parameter(s).

## 5.4 The dialogue module

### 5.4.1 Functionality

The dialogue module is responsible for maintaining the coherence of the dialogue. Thus it is responsible for:

- dialogue structure,
- turn management,
- motivating system utterances,
- predicting the caller’s next utterance,

- interpreting the caller's utterances,
- extending the dialogue history and
- adjusting the dialogue strategy according to the current dialogue success level.

Scheduling and arbitrating requests for output is described in this subsection for the sake of clarity. However, as a matter of implementation, this functionality may belong to the I/O decider.

#### 5.4.1.1 Dialogue structure

At the heart of the dialogue module is a **dialogue grammar** which describes at a fairly high level of generality the range of possible dialogue structures which can be handled by the Sundial system. The dialogue grammar can be thought of as a generative grammar whose start symbol is the general descriptor of any dialogue and whose terminal symbols—at least within the dialogue module—are **dialogue acts**. Partners bring into Sundial expertise in a number of relevant analytic frameworks. For example, UK partners have previously worked within the framework known as Conversation Analysis which presents the notion of adjacency pairs, while French partners bring expertise in Roulet's analysis of tripartite exchanges consisting of *initiative, reaction,* and *evaluation* (see chapter 3). In either case, it is possible to construct grammatical rules which represent dialogue exchanges as tree-structured objects. For example a dialogue might be described at the highest level by the following rule:

dialogue $\rightarrow$ opening body closing

The 'body' of a dialogue might consist of one or more exchanges:

body $\rightarrow$ exchange
body $\rightarrow$ exchange body

One type of exchange might be a 'question-exchange':

question-exchange $\rightarrow$ question answer
question-exchange $\rightarrow$ question question-exchange answer

The first rule in the above example represents a simple question/answer pair. The second rule describes a more complicated pair with an embedded clarification sequence (or sequences). Dialogue acts can be thought of as the terminal symbols of a dialogue grammar in the Sundial system. For a fuller discussion of dialogue grammars see [**?**] and [**?**].

The examples above illustrate the principle of dialogue grammars and do not necessarily correspond in their content to the dialogue grammars to be developed for the Sundial system. A considerable amount of work has already been done towards the identification of the set of required pragmatic functions and the adoption of a common naming strategy for them. These are required to ensure that partners work with a shared set of dialogue acts. A common inventory of dialogue acts will be an important first step towards being able to construct dialogue grammars and make detailed comparisons of dialogue structure between languages. The dialogue grammars themselves will be constructed on the basis of analyses of the simulation corpora (see [**?**]).

#### 5.4.1.2 Turn management

A simple dialogue grammar does not identify who is allowed to speak next. However, this information is crucial. Thus there needs to be an extension to the grammar which identifies who is currently speaking and the permitted next speaker. It is possible that the permitted next speaker will be indeterminate on some occasions. For example, after the system answers a user's question there are two possibilities. Either the user can speak (for example, to ask another question or to close the dialogue) or the system can speak (for example, to supply additional information or to ask if another service is required). The dialogue module must incorporate strategies for dealing with such points of uncertainty. For example, the system will wait until a certain amount of time has elapsed with no speaker (probably between 0.7 and 1 second) and then carry on with the next system utterance (for a discussion of this approach see Sundial WP7 D1).

A different problem which must be dealt with is the case when it is clearly the user's turn to speak (e.g. immediately following a question) and no utterance is detected. In such cases the system must initiate some sort of recovery. Since this kind of problem could occur anywhere in a dialogue (for example, it could result from mishearing, misunderstanding or distraction on the part of the user) there must be some way of keeping a pointer to the present place in the dialogue grammar and entering a temporary recovery phase (see section **??**).

#### 5.4.1.3 Motivating system utterances

It is a well-known platitude that computers will only be able to converse when they have something to talk about. Each Sundial system will have plenty to talk about — after all, each will have access to a large database — but this is not enough. There must be an agenda motivating the system to stop listening and start speaking. Much of this agenda is written in the combined dialogue grammar and strategy for turn management, but the remaining part of it draws upon the task module to identify when a the caller has given sufficient information or, if there is still insufficient information, to identify what is lacking. The dialogue module identifies when the system *may* speak but it must go farther than this and *command the system to start speaking.* Since information involved in the production of an utterance is spread across several modules and sub-modules, responsibility for deciding to initiate an utterance must devolve to one of these: the dialogue module.

#### 5.4.1.4 Predicting the caller's next utterance

An important rôle for the dialogue module is the prediction of the caller's next utterance. This is necessary for two reasons. Firstly, predictions are used to supply top-down constraints to lower levels, thus limiting the search space. Secondly, the predictions are required for interpreting the next utterance once it has been uttered. As we have seen, the smallest unit of analysis referred to in the dialogue grammar is the dialogue act. Since there is a one-to-many relation between dialogue acts and realisations, the predictions must be expected to be fairly weak. If there are also several possible next dialogue acts, the predictions would be further weakened. However, not all dialogue acts are equally likely at a given point so this information can be used to order sets of predictions.

For example, in the French simulation corpus, different dialogue acts appear as responses to the system question “in which class do you wish to travel?” with significantly different probabilities:

| Answer | Answer + Question | Answer + Information | Req. for repetition |
|---|---|---|---|
| 93% | 2% | 3% | 2% |

Clearly there is a strong prediction that the next utterance will be a simple ‘answer’. However, the dialogue module also possesses information concerning task parameters (this information having been supplied by the task module).

If the system asks "in which class do you wish to travel", there is a strong expectation that the user's answer will instantiate the CLASS parameter. Where two values for a parameter are easily confusable (e.g. *Rome* and *Bonn*) and the system clarifies which one is required, the one which is not required should be recorded in the belief module as a dispreferred parameter value, thus ensuring that the rest of the dialogue can make use of the strongest possible predictions.

Chapter 6 discusses the nature of predictions in more detail.

#### 5.4.1.5 Interpreting the caller's utterances

Dialogic interpretation takes place in the context of an ordered set of predictions. Thus it could be argued that much of the task of dialogic interpretation occurs *a priori* in the construction of predictions and at the interface to the linguistic processor where top-down predictions and bottom-up analyses are matched. However, a number of tasks remain: new entities must be introduced; new referring expressions for given entities must be interpreted as alternative formulations; anaphora must be resolved. These tasks are accomplished by the dialogue module in collaboration with the belief module.

#### 5.4.1.6 Extending the dialogue history

The dialogue module is responsible for maintaining the dialogue history. Here, we shall give a brief description of the structure of this history.

The atomic piece of information in the dialogue history is the dialogue act. In one turn more than one dialogue act can be uttered, for example a corrective answer followed by a suggestive answer as in the utterance: "there is no flight at 10.30 but there is one at 11". Thus an object in the dialogue history is of the following format:

$$\begin{bmatrix} parent: & < exchangeID > \\ attention: & < pointer\ into\ the\ belief\ module > \\ locutor: & < user, system > \\ struct-type: & < turn, act, dialogue > \\ act_1: & dact_1 \\ \ldots & \\ act_n: & dact_n \end{bmatrix}$$

The parent slot denotes the exchange in which the turn belongs. The attention slot denotes the attentional states and points to a partition (e.g. a vista) into the knowledge base of the belief module. The $struct-type$ slot is set to “turn” when the object references a turn consisting of several acts, or “act” when it references a single act (in which case $n$ is 1). The outermost object is of ‘dialogue’ structural type.

#### 5.4.1.7 Adjusting the dialogue strategy according to the current dialogue

The dialogue strategy embodied in the dialogue grammar will be designed to be as co-operative as possible and to forestall potential problems. However, problems are bound to occur, for example, system or user mishearings or misunderstandings. These could happen at any point in the dialogue.

A number of approaches to integrating failure and recovery grammars into the main dialogue grammar have been considered. One approach would be to have a set of failure-recovery dialogue acts as possible extensions of every rule in the dialogue grammar. However, this would introduce overwhelming ambiguity into the processing of all routine, non-problematic utterances. The approach to be adopted is to work with the notion of a main dialogue grammar and one or more failure-recovery grammars. When no legitimate next move is found in the set of paths through the dialogue grammar, the current position in the grammar is stacked and control transferred to a failure-recovery grammar. This failure-recovery grammar would embody a different dialogue strategy. If the failure-recovery grammar produced a satisfactory result (i.e. resolved the original problem) the stack would be popped to return control to the main dialogue grammar at the point where it had been left. If however, the problem is not resolved by the failure-recovery grammar, control is passed to a grammar specifying an even simpler dialogue strategy, and so on until a solution or a deadlock were reached. For example, repeated failure on the part of the system to hear or understand a user utterance (it would be difficult to tell which had happened) might lead (i) to a request for repetition, (ii) to a request for rephrasing, (iii) to a request to choose an option from a menu, (iv) to a request to spell an option from a menu.[2].

This approach to dialogue management allows normal unproblematic utterances to be processed in straightforward fashion, whilst the dialogue strategy can be simplified gracefully where it needs to be. The approach embodies notions of global and local management: there is a global plan—specified by the main dialogue grammar—and a number of failure-recovery grammars to manage problems locally. It will be necessary to revise the global plan in the face of persistent problems. For example, a poor quality telephone line might force frequent re-

[2]This progression is illustrative and will be refined through analysis of the simulation corpora

coveries. The dialogue module will monitor the alternations between global and local plans and will have the ability to substitute a simplified (eg. menu-based) global plan if some performance threshold (e.g. $M$ failures in $N$ attempts) is not met.

### 5.4.2 Dialogue module interactions

The dialogue module will send the linguistic interface an ordered set of predictions. It will receive from the linguistic interface an instantiated predication. It will send out messages to the message planning module. These messages will consist of a dialogue act(s) with a deep representation of the semantic content of the act, such as "get(departure_time)"[3] to ask the user about the value of the departure time parameter. The dialogue module will need to interact with the belief module in order to interpret utterances. It must also interact with the task module to obtain lists of parameters to build into queries to the user. It must return parameter lists to the task module for it to build into database queries.

## 5.5 The belief module

### 5.5.1 Functionality

The module is responsible for maintaining a representation of objects and relations in the domain; these may exist by virtue of having been mentioned, or via inferences and defaults. Other modules may treat it as a database, to which requests can be sent, e.g.:

- is this object unique?
- is this object reliable?
- is this object in focus?
- what is an appropriate description for this entity?

This module will keep separate the information coming from the user and from the system in order to record the origin of the knowledge.

[3]this notation is used here only for the sake of clarity.

#### 5.5.1.1 Focus maintenance and Referential interpretation

The belief module is responsible for the adequate storage and retrieval of the objects that the participants of the dialogue have at one time or other talked about. Its task is also to keep information about the order in which these objects have been referred to and about which single object or group of objects is at the centre of the interaction, i.e. in focus. The order in which objects are introduced is implicitly contained in the dialogue history. As a means of representing objects in a focus space we will use partitioning techniques [**?**] [**?**]. In the dialogue history we will have pointers to partitions ("vistas" in Hendrix' terminology).

When the dialogue module decides to introduce a new topic it sends a message to the belief module to create a new focus space, which will become the current (active) one. Apart from this being necessary for the resolution of anaphora and ellipses, it is also necessary for the maintenance of dialogic coherence, ensuring that at any moment user and system are talking about the same objects. If the interpretation of the user utterance needs references outside the focus space then the belief module has to advise the dialogue module. This facilitates the interpretation of interruptions (in Grosz' terminology).

#### 5.5.1.2 Task specific interpretation using word/domain knowledge

The belief module is responsible for translating the surface semantic representation coming from the linguistic/semantic analysis into a deep structure understandable by the task module, that is, a deep representation which could answer task goals such as

"get_parameter(arrival_city)".

This transformation could be very simple: for example when the user says "I want to go to Paris", then the arrival city slot in the reservation frame would be instantiated with the value "Paris". This interpretation process should keep tracks of what modality has been used and what constraints have been uttered. For example, "I want to leave *at* 10.30" and "I want to leave *around* 10.30" should have different deep representations.

During the interpretation process the belief module should test the acoustic reliability of what has been said and make available to the dialogue module its acoustic status.

### 5.5.2 Belief module interactions

This module receives representations from the linguistic interface module which are 'ungrounded', in that eventual reference resolutions are to be done. It will send deep representations of what the user said to the task and dialogue modules. It will reply also to requests about focus information and requests from the task about the user's requirements. It will also make available acoustic status reports on values in its knowledge base.

## 5.6 The linguistic input interface module

This module is responsible for maintaining communication channels with the linguistic processing and front end components. The main functions are:

- reduce multiple hypotheses, pack them, or choose from amongst them;
- organise recovery from total or partial analysis failure;
- channel predictions emanating from other dialogue modules to the parser.

This module is also responsible for maintaining the linguistic history, which consists of the linguistic productions of each speaker. The module will place each surface utterance from the linguistic processor into the history. Also, when the message planner has decided on a production, it will send the result to the linguistic interface to add it to the linguistic history. Conceptually, this history is shared by both the message planner and the linguistic interface and is therefore located between these two modules in figure **??**.

## 5.7 The message planning module

The message planning module's main functions are:

- to construct a global plan which expresses the semantic content of the system message to be conveyed to the user;
- to construct local plans for expanding references to referring expressions, in response to requests made by the Message Generator.

The module is therefore responsible for managing a bi-directional flow of information between the Dialogue Manager and the Message Generator.

# Chapter 6

# Knowledge Representation and Interfaces

## 6.1 Introduction

In this chapter, we describe the interfaces between the dialogue manager module (henceforth, DM) and the linguistic processing (LP) and message generation (MG) modules. The interfaces between these modules respect the Distributed Database Architecture[1]. One important aspect of these interfaces is that they are *bi-directional*. With the LP–DM interface, a prediction about the incoming utterance is passed from the DM to the LP and a linguistic representation of the current utterance is passed up to the DM. With the DM–MG interface, the DM sends a global plan of the next system utterance. The MG can send requests to the DM for further semantic information and, when linguistic realization is complete, returns a linguistic representation to the DM.

Given the relative complexity of these interfaces, we have decided to develop a common language, the *Semantic Interface Language* or SIL, to express messages between modules. This should, to a large extent, obviate the need for translation processes which convert representations used in different modules. It should also facilitate the construction of interchangeable software modules: i.e. it will sufficiently restrictive so that modules which follow different design strategies in, say, linguistic processing and message generation, can still be interfaced with other modules in the system.

While the language was developed primarily for specifying communication between the Dialogue Manager and other components of the SUNDIAL system, it

[1]This approach to architecture is described in deliverable WP2, D1.

will also be useful in specifying the information flows between sub-modules within the Dialogue Manager and will form the basis of the representation used in the various knowledge bases held by each sub-module. It therefore has some of the characteristics of the knowledge representation languages discussed in Chapter 4.

At this stage, we have agreed on some, but not all, aspects of the language. The language itself is a functional rather than implementation language: it specifies the content and structure of messages without restricting how these messages are to be coded[2]. Extending the notion of a *sign* in constraint-based natural language processing (cf. [**?**]), information about utterances is classified along four dimensions: dialogue, semantics, syntax and phonology[3].

$$\begin{bmatrix} \begin{bmatrix} DIALOGUE \end{bmatrix} \\ \begin{bmatrix} SEMANTICS \end{bmatrix} \\ \begin{bmatrix} SYNTAX \end{bmatrix} \\ \begin{bmatrix} PHONOLOGY \end{bmatrix} \end{bmatrix}$$

Of these dimensions, it is **semantics** which has received most attention so far. There are two reasons for this. Firstly, semantic information is the information which most affects the behaviour of the importing module. With the LP–DM interface, the LP will need to specify the meaning of an utterance since it is this information which primarily drives the belief, task and dialogue modules in the DM. Likwise, most predictions from the DM will include a semantic description of the upcoming utterance. With the DM–MG, the global plan sent to the LP will include a (partial) semantic description of the next system utterance and the requests sent from the LP to the DM seek further semantic specification of parts of the utterance. Secondly, with semantic information we should be able to minimize language dependences: i.e. whereas the other types of information may be language-particular, we assume that conceptual content and structure can be described so as to be equally applicable to French, German, Italian and English. For these reasons, we have devoted much effort to the development of a common semantic representation language which can be used for both interfaces.

However, some aspects of the other types of information have been agreed upon. The representation of **dialogue** information will be based upon an agreed representation for dialogue acts and include dialogue features to be used by the prosodic component in the MG. The **syntax** dimension will include a parse or analysis tree for the utterance and will include morpho-syntactic features, such

[2]Error messages and communication protocols, for example, are not covered yet.

[3]The phonology dimension should not be taken as a detailed representation of phonological information in the traditional sense. Rather it is an (augmented) orthographic representation of words and word chains.

as [*inverted*] and [*mood*] marked on the syntactic categories. The **phonology** dimension will contain a string representation of the segmental and suprasegmental phonology of the utterance, including prosodic features, such as [$question - intonation$] and acoustic scores[4].

### 6.1.1 Semantic Interface Language

The following description of the semantic interface language (SIL) should be seen as a preliminary version which will be revised and extended on the basis of partners' experience. We have already identified and discussed further representational issues which we are presently investigating (see section **??**). The description is primarily given from perspective of the LP–DM interface since the range and complexity of utterances interpreted by the system will be greater than those generated by the system.

#### 6.1.1.1 Requirements

There are three general requirements[5].

- **clarity**
- **partiality** the language should be capable of representing partial information
- **expressiveness** the language should be capable of representing analyses of all semantic phenomena encountered in different languages and application domains

Semantic phenomena which might prove difficult to analysis and represent include

- **questions** especially Wh-questions
- **sets of individuals** as well as operations on these sets; for example, *three of the flights you requested ....*
- **attitude verbs** such as *veux*, *desire*, *want* and *exist.*
- **discourse markers** such as *yes*, *thank you* and *well.*

[4]An alternative is that scores may be placed in the semantic dimension.
[5]Requirements on knowledge representations in general are described in chapter 4.

- **3rd person references to traveller** where the user calls on behalf of someone else.
- **degrees of conviction** such as *I think it leaves from London at 10pm*
- **temporal references** such as *now*, *this afternoon*, *tomorrow morning* and *after 3pm.*
- **spatial references** including the *here* and *there*
- **multiple co-ordinations**
- **idiomatic expressions** as in *three hundred and twenty five pounds – that's daylight robbery*
- **vague information** such as *around 7pm*, *I want the last train tonight* and *sometime after 2pm.*
- **calender expressions** such as *the flight on February 23rd*
- **ordinals**
- **places and organizations**
- **codes** such as telephone, flight and train numbers
- **names of individuals**
- **copula sentences**
- **tense**
- **modality**

#### 6.1.1.2 Syntax

In developing SIL, partners drew extensively on their experience working with contemporary semantic representation languages. The most prominent of these were Situation Schemata (cf. [?] and [?]) and Conceptual graphs (cf. [?]). By and large, these languages are expressively and notationally equivalent: i.e. what can be expressed in Conceptual Graph notation can also be expressed in Situation Schemata and vice versa[6]. As a hybird language, SIL is largely compatible with

[6]One difference concerns embedding. Conceptual graphs have far 'flatter' representations of complex objects than those situation schemata. The representations in SIL are more akin to situation schemata in this respect. We are currently evaluating whether this style of representation is suitable for imposing constraints on parsing.

both languages: representations in that the SIL notation can converted into Conceptual Graphs or Situation Schemata notation and vice versa. However, SIL differs in important respects from both languages. Many of these emerge from the representational needs of a task-driven, speech-based information system.

The basic representational unit in SIL is an **object**. Objects are structured representations: semantic information is represented using attribute value pairs. The values of attributes can be atomic or compound and structure-sharing is permitted. The labels for attributes and values are intended to be theory-neutral.

Objects have the following attribute-value structure

$$\begin{bmatrix} \\ ID : \ldots \\ OBJNAME : \ldots \\ TYPE : \ldots \\ MODE : \ldots \\ CARD : \ldots \\ ARG{*} : \ldots \end{bmatrix}$$

The following constraints are placed on the values of these attributes.

The **ID** attribute specifies an index for the object. The value of this attribute is a unique integer which is enclosed in brackets (for notational convenience). If co-indexing is required, as in the case of infinitive complements or anaphoric expressions, the value of an ARG attribute is shared with the index of an object rather than described as a separate object.

The **OBJNAME** attribute names the object. At present this is simply the orthography of the linguistic expression. The relationship between the values of OBJNAME, TYPE and CASELABEL is discussed below.

The **TYPE** attribute specifies the sortal type of the object. Sortal values are nodes in a sortal hierarchy compiled out of the conjunction (and, perhaps, disjunction) of sortal features. One of the functions of this attribute is to constrain syntactic combination (cf. selectional restrictions). At present, the TYPE of a object is simply the orthographic form of the linguistic expression.

The **MODE** attribute specifies a quantificational constraint interpretation of the object. The possible values are drawn from a set of quantifier labels which include *THE*,*A*, *ALL*, *NO*,*?* (for wh-elements in questions), *THENEXT*, *THE-LAST*, *THEFIRST*, *THENTH* (where *N* is an ordinal), *PREVIOUS*, *THEOTHER* and *ANOTHER*. The set of quantifiers will be extended on the basis of partners' actual needs; for example, *?MANY* may be used to represent *how many* and adverbs may be represented as quantifiers over verb predicates. Matters of quantifier scope are not currently addressed in SIL.

The **CARD** attribute specifies cardinality of objects. With entities it can have one of two values: *INDIVIDUAL* or *SET*. The latter value is used for objects whose cardinality is greater than or equal to two. SET values can be further specified, i.e.

$$\begin{bmatrix} SET \\ NUMBER : n \end{bmatrix}$$

where $n \geq 2$. The elements of sets are generally not represented in SIL (at least in the LP–DM interface); objects with set values can be expanded when they are anchored within the belief model of the dialogue manager. When this occurs, the number of elements is constrained to be no more than cardinality of the set.

Each object has an arity represented by a number of **ARG** attribute-value pairs. Objects whose arity is zero are *atomic* objects; those with an arity of one or more are *compound* objects. Simple entities are represented by atomic objects, whereas relations and complex entities are represented as compound objects. Accordingly, the attribute **ARG** is a complex value consisting of attribute-value pairs of the form

$$\begin{bmatrix} CASELABEL : OBJECT \end{bmatrix}$$

where **CASELABEL** is either a case role label such as *AGENT*, *PATIENT* and *LOCATION*, or *DESC* (for description) used for arguments whose role is unspecified[7]. Case role labels are used to represent the argument types of verbs and description labels the modifiers of nouns. The value of the **CASELABEL** attribute is itself an object.

#### 6.1.1.3 Examples

For notational convenience, attribute-value pairs not relevant to the semantics of the expression are omitted.

[7]These case role labels are purely illustrative: we have to finalize the set.

(6) *I want to go to Paris*

$$\left[\begin{array}{l} ID : [1] \\ OBJNAME : want \\ TYPE :< want > \\ \\ AGENT : \left[\begin{array}{l} ID : [2] \\ OBJNAME : I \\ TYPE :< I > \\ CARD : INDIVIDUAL \end{array}\right] \\ \\ OBJECT : \left[\begin{array}{l} ID[3] \\ OBJNAME : go \\ TYPE :< go > \\ \\ AGENT : [2] \\ \\ DEST : \left[\begin{array}{l} ID : [4] \\ OBJNAME : Paris \\ TYPE :< Paris > \end{array}\right] \end{array}\right] \end{array}\right]$$

A number of observations can be made about (6)

- the TYPE of objects is not specified since we have yet to agree upon a language-independent sortal hierarchy. Once this has been agreed, the sortal type for the agent role of the relation *want* could be $< person >$ and the sortal type for *I* could be $< hearer >$ given that the lexicon-grammar is capable of making indexical interpretations. The latter sortal type subsumes the former type. As for its eventual sortal type, one possibility is that the most specific sortal type remains – in this case, $< hearer >$.

- the second argument of *want* is a compound object since it has specified ARGs.

- the agent of *go* is specified via the co-indexing with the first argument of *want*.

- if this sentence was negated, i.e. *I don't want to go to Paris*, the representation would be identical except that the MODE attribute of *want* would have the value NO.

(7) *. . . the town that I mentioned . . .*

$$\left[\begin{array}{ll}
ID : [1] & \\
OBJNAME : town & \\
TYPE :< town > & \\
MODE : THE & \\
CARD : INDIVIDUAL & \\
DESC : & \left[\begin{array}{ll}
ID : [2] & \\
OBJNAME : mention & \\
TYPE :< mention > & \\
AGENT : & \left[\begin{array}{l}
ID : [3] \\
OBJNAME : I \\
TYPE :< I > \\
CARD : INDIVIDUAL
\end{array}\right] \\
OBJECT : [1] &
\end{array}\right]
\end{array}\right]$$

With example (7) the restrictive relative clause is treated as a unspecified argument of the object *the town.* This analysis also holds for adjectives and other nominal modifiers of nouns.

(8) *. . . a plane which leaves after 13-20*

$$\left[\begin{array}{ll} ID : [1] & \\ OBJNAME : plane & \\ TYPE :< plane > & \\ MODE : A & \\ CARD : INDIVIDUAL & \\ DESC : & \left[\begin{array}{ll} ID : [2] & \\ OBJNAME : leave & \\ TYPE :< leave > & \\ AGENT : [1] & \\ TIME : & \left[\begin{array}{ll} ID : [3] & \\ OBJNAME : after & \\ TYPE :< after > & \\ TIME : & \left[\begin{array}{l} ID : [4] \\ OBJNAME : 13-20 \\ TYPE :< 13-20 > \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

With (8), the TIME role of *leave* is a compound argument which consists of the one-place predicate *after* whose argument has the TIME role (this may be too specific). The representation for *13-20* could be further specified making clear the hour and minutes.

(9) *pouvez vous repeter l'heure de départ*

$$\left[\begin{array}{ll}
ID : [1] & \\
OBJNAME : pouvez & \\
TYPE :< please > & \\
 & \\
AGENT : & \left[\begin{array}{l} ID : [2] \\ OBJNAME : vous \\ TYPE :< system > \\ CARD : INDIVIDUAL \end{array}\right] \\
 & \\
OBJECT : & \left[\begin{array}{ll} ID : [3] & \\ OBJNAME : repeter & \\ TYPE :< repeat > & \\ & \\ AGENT : [2] & \\ & \\ OBJECT : & \left[\begin{array}{l} ID : [4] \\ OBJNAME : l'heurededépart \\ TYPE :< departuretime > \\ MODE : THE \\ CARD : INDIVIDUAL \end{array}\right] \end{array}\right]
\end{array}\right]$$

In example (9), taken from dialogue 202 of the French simulation corpus, the sortal type for *pouvez* will reflect its discourse function. Discourse markers, such as *yes* and *well*, will also be TYPEs that reflect their discourse function and they will be represented in objects separate from those which specify referential information (as with *yes, I want* ...).

(10) *combien de places voulez vous reserver*

$$\left[\begin{array}{l} ID : [1] \\ OBJNAME : voulez \\ TYPE :< want > \\ \\ AGENT : \left[\begin{array}{l} ID : [2] \\ OBJNAME : vous \\ TYPE :< system > \\ CARD : INDIVIDUAL \end{array}\right] \\ \\ OBJECT : \left[\begin{array}{l} ID : [3] \\ OBJNAME : reserver \\ TYPE :< reserve > \\ \\ AGENT : [2] \\ \\ OBJECT : \left[\begin{array}{l} ID : [4] \\ OBJNAME : places \\ TYPE :< seat > \\ MODE :?MANY \\ CARD : SET \end{array}\right] \end{array}\right] \end{array}\right]$$

With (10), the representation of the question is highly tentative and not fully agreed upon. The fact that it is a question is not marked at the top-level, but only on the argument about which information is being sought. An analogous treatment of 'yes-no' questions would have the MODE of the main verb assign the value *?*. An alternative treatment involves a top-level object which specifies that the utterance is a question, with an 'Wh' argument co-indexed to one of the arguments embedded in the representation of *voulez*.

### 6.1.1.4 Future Extensions

Our eventual aim in extending SIL is give it a formal syntax and semantics. In order to achieve this, we have identified two major tasks: to develop precise constraints on the possible values of the attributes; and to specify the interpretation of this notation within the dialogue manager. In doing this, we need to resolve issues concerning the interpretation of various phenomena so far not covered by SIL as well as the extent to which the linguistic processing and message generation are capable of making task-dependent interpretations.

It should be obvious that much work still needs to be done on specifying the possible values for the attributes of objects. In particular, we are working on tighter definitions of the OBJNAME, TYPE, MODE and CASELABEL attribute-value pairs. With OBJNAME and CASELABEL, we still have to decide whether these can affect how SIL expressions are anchored within the dialogue manager (i.e. the belief model). If we follow a traditional approach where TYPE is just used to constrain syntactic combination and the OBJNAME to constrain interpretation, then the information specified in the OBJNAME, TYPE and CASELABELs of objects will, to some extent, be redundant. For example, with a verb like *go*, the number and nature of the arguments will be expressed in two ways: by means of their sortal type or by means of the number and nature of the argument roles. Accordingly, we are presently assessing the trade-off between using the TYPE specification vis-a-vis OBJNAMEs and CASELABELs for constraining interpretation as well as syntactic combination in linguistic interpretation and realization. Part of this involves the development of a fully specified sortal hierarchy which, at the top and middle levels of the hierarchy at least, is applicable in all languages and application domains. One possibility is to use a hierarchy like that developed by McDowell [**?**]. We are also developing a set of meta-constraints, constraints which specify the relation between elements in the sortal hierachy and the semantics of lexical items. Finally, just as with TYPE, we expect to develop a feature logic for MODE in which we specify all the well-formed combinations of quantifier features. Such a logic will take on board the reality of operating with a speech-based information system; for example, since the front-end may be unable to differentiate between the articles *the* and *a*, we need a MODE value which covers this eventuality.

Other extensions to the notations are directed at increasing its expressive power. Our current agenda of difficult phenomena will be progressively covered by the notation although the agenda itself may be increased on the basis of our experience.

The interpretation of SIL representation takes place within the belief module in the DM. From the perspective of the LP–DM interface, we see utterance interpretation as taking place in at least two phases (cf. Bunt *et al* [**?**])

- a *context-independent* interpretation is produced by the parser, according to compositional rules specified in the grammar/lexicon.
- a *context-dependent* interpretation is produced by the dialogue manager: i.e. the representation produced by the LP is anchored to entities and relations in the belief model. This anchoring takes into account the context of the utterance and results in the extension of the current belief model.

Only some issues in interpretation have been agreed upon. One is that interpretation will not be truth-conditional: more likely, it will involve evaluation of contextually-extended SIL representations against the situations described in the belief model. Secondly, objects will be extended within the context of the current belief model. In the case of referring expressions, they will be anchored to the appropriate entities within the belief model. Part of this anchoring process may involve the assignment of a *predecessor* index which relates the current description of an entity to previous descriptions in the belief model. This allows us to track the derivational history of objects as the belief model evolves[8]. In the case of attitude verbs, they may be anchored within the relevant partitions of the belief model; for example, utterances expressing a 'want' on the part of the user will be anchored withn the *user-wants* partitions.

There are two principal reasons why interpretation has not been finalized. One is that we have yet to specify the structure of the belief model. Another concerns the extent to which linguistic processing will make 'contextual' interpretations; in particular, the extent to which compositional rules in linguistic processing can be used to determine anaphoric and task interpretations[9]. Clearly some anaphoric co-indexing can be achieved outwith the DM; for example, the anaphoric co-indexing of the 'missing' subject in infinitival complements of verbs like *want*. However, other cases of anaphoric linking would require contextual information not in the utterance being interpreted and this information is not directly available to the LP or the MG. Since the linguistic history in the linguistic interface module of the DM records this information, the LP could request the information necessary to resolve anaphora[10]. Alternatively, most anaphoric links could be established in the DM directly using the linguistic history. Various strategies for establishing anaphoric links are currently being explored and the various trade-offs are being evaluated[11]

The same approach has been adopted to task-specific interpretation. One strategy is to maximize task-specific interpretation outwith the dialogue manager. Task-specific information can be hardwired into the grammar lexicon: i.e. task-specific information is included in the lexical semantics of task-related expressions. The advantage of this strategy is that it considerably reduces the amount of task-specific interpretation within the dialogue manager. An alternative is not to include task-specific information in the lexicon and instead perform all task-specific interpretation within the dialogue manager. The advantage of

[8]One advantage of this is that it should facilitate belief revision.

[9]The determination of quantifier scope is another. The extent to which this issue bears upon the Sundial system is unclear at present. However, we could draw upon research on Discourse Representation Theory and Situtation Semantics undertaken by another ESPRIT project DYANA.

[10]This would occur if incremental evaluation of referring expressions takes place. This should still allow for a declarative definition of the interface.

[11]This includes the extent to which the trade-off may be language dependent; languages with rich morphological system, such as German, may facilitate 'early' interpretation.

this is that, in principle, the same lexicon can be used without modification in different task domains. The difference between these strategies can seen with the interpretation of the expression *seat* in utterances such as *I want to reserve a seat, a return, Paris-London, 23rd February.* Adopting the first strategy would result in lexical entry for *seat* which was approximately equivalent to *flight* and, in the case of a return flight, would include roles for the outward flight and inward flights, each with their own attribute value pairs for the time, date, location, city and so on. Information in the post-modifier phrases could then be integrated into the main representation prior to interpretation in the DM. With the second strategy, *seat* would lack these argument roles and the information that it is a return, the departure and arrival locations for the outward flight and the date would be represented as separate objects in the representation. Only when the representation was interpreted in the dialogue manager would these be attached as arguments of a 'flight' predicate. While we generally favour the first strategy, we still need to assess the degree to which all task-specific interpretation can be satisfactorily dealt with in this way or whether the interpretative strategy might vary with task complexity and domain.

## 6.2 Interface to the Linguistic Processing Module

Having discussed some aspects of the language in which inter-module messages are expressed, we now turn to more specific matters concerning these interfaces. With both interfaces, one of the modules within the DM acts as the 'primary' interface module. With the LP–DM interface, this is the Linguistic Interface module and with the DM–MG interface this is the Message Planning module.

### 6.2.1 Linguistic Interface Module

#### 6.2.1.1 Objectives

The function of the linguistic interface module (LI) is to mediate interaction between the linguistic processing module and the rest of the dialogue manager. The module has the following functionality

1. It constructs predictions on the basis of information provided by other modules in the DM.

2. It updates and maintains the linguistic history of the system. The linguistic history records utterances made by the user and system in the form of their linguistic representations.

3. In the event of incremental interpretation, it replies to requests from the LP for anaphoric and ellipsis resolution.

#### 6.2.1.2 Principles

The simplest relationship between the LP and the linguistic interface module (LI) is a unidirectional information flow: the parser constructs a linguistic representation of the utterance which is sent to the LI. The semantic objects in this representation are then anchored in the belief module and the linguistic history is updated. In the Sundial system, however, there is a bi-directional information flow: the DM, via the LI, sends the LP predictions which guide the linguistic interpretation of the utterance. The intended advantage of predictions is to increase the computational efficiency and robustness of linguistic processing.

Predictions are generated by merging information available within the dialogue manager: in particular, dialogic, belief and task goals. The merging of these goals is performed by the LI[12]. These predictions are available to the LP in the form of expressions in the interface language. As we said earlier, these specify four types of information: dialogic, semantic, syntactic and phonological information. While one might expect predictions about the dialogic function of upcoming utterances to be specified in terms of dialogic information, this is not the case since we assume that the LP is incapable of interpreting dialogue act information[13]. Instead, the dialogic goals are converted into predictions about other levels of linguistic structure: i.e. semantic, syntactic and phonological information. For example, if the dialogue module predicts that the next user's utterance will be a 'yes-no' question, this prediction has to be ground out in terms of the syntactic structure of the utterance, perhaps using the syntactic feature [+inverted] on the top-category; the semantic structure, using the value *?* for the top-level object's MODE attribute; and the phonological structure, using phonological features such as [+question-intonation]. If the prediction is sufficiently specific then this

[12]Another possibility we are exploring is that merging can be performed by the input/output decider (see chapter 5). The decider would not only prioritize goals for message generation, but merge and order goals to be sent to the LP as predictions and to the MP dialogic plans.

[13]This assumption may be revised in light of attempts to build a parser which construct charts using dialogic information. It also highlights an asymmetry between the linguistic processing capacities of the linguistic interpretation and generation processes since the latter is assumed to make linguistic decisions on the basis of dialogue act information. This seems to be a weakness with the interpretative side of the Sundial system, a weakness that is also manifest in its failure to deal with prosodic information.

will result in the generation of a sub-grammar: i.e. a set of permissible syntactic rules together with a set of permissible expressions (either directly in terms of particular lexical items or constraints on their sortal TYPE).

This conversion from semantic information to other types of linguistic information could, in principle, be part of the functionality of either the LI or the LP. Our tentative decision to make it part of the functionality of the LP is based upon the assumption that the DM does not have access to grammatical representations and rules[14]: i.e. just as dialogic information is outwith the scope of the LP, so grammatical information is outwith the scope of the DM. On the other hand, the LI does have access to the linguistic history and hence specific linguistic information can be incorporated in predictions where the user is expected to 're-use' previous expressions. For example, if the user has been asked to confirm the value of a parameter, such as the arrival time, then the prediction would include specific grammatical information recorded in the linguistic history. In most cases, predictions will simply specify semantic information. For example, if the system generated the message *can you tell me where BA 285 leaves from?*, the prediction would include the information that there will be an object whose TYPE is restricted to that of locations. Further elaboration of this prediction could be performed within the LP: i.e. it could include the possible syntactic realizations and, perhaps, the set of lexical items expressing particular locations (e.g. New York, Paris, etc). However, we are aware that predictions will not always facilitate linguistic processing: generating one or more 'vague' predictions may hinder, rather than help, linguistic processing, especially since its other input is a lattice. In such cases, construction of a linguistic representation might be best performed in a bottom-up manner. The trade-off between predictive specificity and facilitation of linguistic processing is currently being explored by experimenting with various strategies.

The LI generates a single prediction for the next user utterance. Thus rather than the LP making repeated calls to the DM for predictions, only one prediction is generated. However, this prediction can be 'packed' and the alternatives within it ordered. Packing is performed by the LI: rather than assign a unique linguistic description of the utterance, the utterance is assigned a disjunctive description. For example, the DM may predict that user will describe the departure location as either *Paris* or *London*. One possible restriction on packing is that packing will be 'local' (see below). Furthermore, alternatives within predictions can be assigned a priority order; for example, *London* may be a more 'typical' departure location than *Paris*. Assigning priority ordering may be extended to the generation of a sub-grammar: i.e. syntactic realizations of the semantic information may differ in priority. In both these cases, the priority ordering will be empirically determined.

[14]The same holds for the message planning module.

In addition to generating predictions, the other major function of the LI is to update and maintain the linguistic history. This history is updated on the basis of information from two modules, the LP and MG. Furthermore, while the representation from the LP may be a 'locally' packed structure (see below), it contains only one analysis of the utterance. When the LP generates more than one analysis of an utterance, only one is passed up to the LI for inclusion in the linguistic history. Which one is selected may depend upon global acoustic scores and/or various scores for structurally ambiguous utterances. As for the output of MG, the linguistic representation of the utterance is sent via the message planning module to the LI and updates are performed on other histories, such as the belief and dialogue models. Accordingly, we envisage neither forced acoustic recognition nor forced linguistic and dialogic interpretation of system utterances. Once in the linguistic history, representations may be accessed in two ways: by means of 'span' indices which are assigned to its constituents when they are either analysed by the acoustic front-end or generated by the speech-synthesizer; or by means of 'anchor' indices which anchor the semantic representations to entities and relations in the belief model. If the notion of 'predecessor' indices are accepted as part of the structure of the belief model – i.e. indices which record the derivational history of a given object – then the DM is able to trace the different linguistic realization of the same object via the anchor indices in the linguistic history. Finally, it is the linguistic history which allows the DM to resolve cases of local as well as global anaphora. This will allow the LI to reply to requests from the LP for anaphora and ellipsis resolution when interpretation is incremental.

### 6.2.2 Information flows across the LP–DM Interface

We have described the nature of the bi-directional information flow at the LP–DM interface in terms of interactions between the linguistic interface module in the DM and the LP. The LP passes the DM a linguistic representation of the incoming utterance. This representation has been constructed not only on the basis of information from the acoustic front-end, but also on the basis of predictions from the DM. These predictions are constructed on the basis of belief, task and dialogic goals and these guide the LP's construction of the linguistic representation. This interface is shown in figure **??**.

### 6.2.3 Further Issues

This description of the interface has not addressed all issues bearing upon information flow between these modules. In this section, we discussed further issues

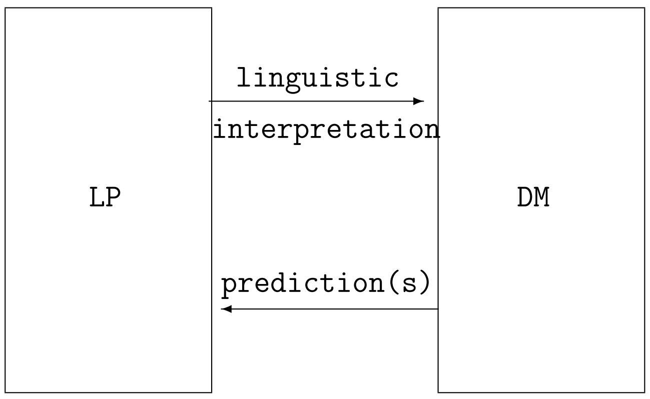

Figure 6.1: The LP–DM Interface

which we have identified as relevant to the definition, but not yet fully resolved.

1. **multiple LP analyses** As it stands, the interface language makes no allowance for cases where linguistic processing constructs competing representations (or hypotheses) for the same span in the lattice. Such cases range from representations which only differ with respect the interpretation of a single sub-span of the lattice (roughly, a single word) through those where there are multiple interpretations for many sub-spans of the lattice to those where the lattice as a whole is assigned competing linguistic representations. In advocating that only a single representation was passed up from the LP, we allowed for the possibility that this representation may be 'locally' packed. By local packing we mean that multiple hypotheses for the **same** argument or relation elements are represented. That is, if two hypotheses only differ on the representation of span in the lattice which is analysed as a semantic argument or relation, then these are represented as disjunctive elements in the representation. For example, in constructing a linguistic representation for *When does it arrive at Lannion?*, linguistic processing may have difficulty in deciding between *Lannion* and *Toulon* as the appropriate analysis for the final sub-span of the lattice and thus the corresponding *location* argument in the semantic representation. To choose one of the alternatives arbitrarily, or on the basis of a marginally higher confidence score, may well prove wrong. However, if the alternatives are packed (the parser can do this as part of the normal business of chart construction), then the dialogue manager will be able to make informed decisions about how to proceed. For example, in some cases, one alternative might be choosen on the basis of (in)compatibility with previous descriptions of the arrival location. More usually, when a unique anchoring in the belief model is unavailable, there will be a anchoring failure on a task-related

object and this will cause the dialogue manager to plan a 'directed' confirmation request — for example, *did you say Lannion or Toulon?*. After this conflict has been resolved, the dialogue manager may then send a 'bias' to the acoustic front-end (via the LP) so that the conflict does not arise again.

2. **partial and total failure in the LP** In some cases linguistic processing will identify some but not all spans in a lattice. In cases where there is a partial failure which correlates with an argument or relation in the semantic representation, the failure can be flagged by linguistic processing, perhaps with a query value for the relevant argument. The dialogue manager can use this partial representation to either infer an argument or generate a more 'intelligent' recovery message – for example, *what time did you say the flight leaves?* If there are multiple failures or total failure, then the message from the LP would simply be a failure flag which would cause the dialogue manager to seek repetition or re-phrasing of the whole utterance.

3. **competing DM predictions** One approach parallels that adopted with multiple LP analyses: only one prediction, possibly with local packing, is passed down. The problem with this strategy is that DM predictions are not necessarily compact. For example, in the French simulation users made the following responses to the system question *how many seats do you want to reserve?*

   | | |
   |---|---|
   | 90.7% | <number><seats> |
   | 7.9% | <number><seats> plus new information |
   | 0.8% | <number><seats> plus question |
   | 1.16% | request for repetition |

   If only local packing was permitted, then only the first of these predictions would be sent to the LP. Another approach is to allow global packing in predictions: i.e. a prediction consists of multiple linguistic representations (as above) which may share little or no semantic structure, but cover the same linguistic span. These representations could be assigned a priority ordering on the basis of simulation results. A third alternative is that these are treated as individual predictions and sent down one by one following the priority order. If the first prediction failed, linguistic processing could request next best prediction. One worry about the third approach is that successive, conflicting predications might disfacilitate rather than facilitate linguistic processing.

4. **exhaustive predictions** If predictions are exhaustive, then the representation generated by LP must be subsumed by at least one prediction. In this way, the DM would generate a 'hollow' linguistic specification which is filled in by the LP. Exhaustive predictions could have narrow or wide scope: with

wide scope they would cover all possible user utterances; and with narrow scope they would cover only a subset of utterances. Wide scope exhaustive predictions are clearly not feasible, but narrow scope predictions would effectively restrict the permissible responses to a 'menu' style interaction. This, while more practical from our point of view, will inevitably impinge upon the 'naturalness' of the dialogue. Of course, the alternative is that predications are non-exhaustive: dialogue manager predictions cover some but not all utterances. The implication here is that linguistic processing, when available predictions fail, might still be able to construct a representation for the utterance. In this case, the output of linguistic processing would not subsume a prediction, but lie outside it. One strategy would be build the first demonstrator on the basis of narrow scope predictions and explore the possibility of non-exhaustive predictions for the second demonstator.

5. **predictions as filters** Treating predictions as filters, the issue here is how far these filters are to be pushed down into linguistic processing. We could push them deep into linguistic processing so that they drive the construction of linguistic representation; if one of these constraints were violated, then a representation would not be built. Alternatively, predications could simply filter linguistic processing: representations are constructed without reference to predictions and only when a representation has been constructed are they compared.

6. **Incremental versus Batching interpretation** The problem with incremental interpretation is that if 'uncertain' semantic representations are grounded in the belief model and these turn out to be wrong, then a large amount of processing time might be spent repairing the belief model. If incremental interpretation only operates on the basis of 'certain' semantic representations, then it is not clear what advantage it would have over batch processing.

## 6.3 Interface to the Message Generation Module

In this section we describe the interface between the DM and the MG. Interaction between these modules parallels the LP–DM interface in so far as they are bidirectional interfaces and messages passed between the modules are expressed in the same interface language. The main difference concerns interactional complexity since the MG sends request to the message planning module (MP) to further determine the description of referential expressions. This difference disappears if the LP can incrementally request anaphoric and ellipsis resolutions.

### 6.3.1 Message Planning Module

#### 6.3.1.1 Objectives

The MP has two objectives

- it constructs a global plan which the linguistic realization module (LR) in the MG attempts to realize
- it constructs a local plan for referring expression in response to requests from the LR within the MG.

#### 6.3.1.2 Principles

Message generation has been most frequently studied within the context of text rather than speech generation. There message generation is conventionally split into two stages: planning and realization. Although the boundary between these stages is not clear cut (cf. [?]), many researcher take planning as the determination of the **non-linguistic content** of the message and the realization as the determination of its **linguistic form**. Thus [?] and [?] talk of planning as a *strategic* function and realization as a *tactic* function. [?] makes an analogous distinction between deciding *what to say* and deciding *how to say it* and [?] of *deep generation* and *surface generation.*

Three major approaches to planning and realization have been adopted in text generation (cf. [?] and [?]). The first, the **integrated approach**, treats planning and realization as different aspects of a uniform process. This approach is manifest in early (pre-TELEGRAM) versions of KAMP (cf. [?], [?] and [?]). With this approach, linguistic realization (and thereby linguistic information) is not localized in one generation stage, but spread throughout the system in procedures and 'critics'[15]. In the second approach, the **sequential approach**, planning and generation are seen as separate processes, or modules, with a *uni-directional* dataflow: i.e. the planning process constructs a plan which is exported to the linguistic realization process. The approach to generation can be seen in [?], [?], [?], [?], [?] and [?]. Whereas the integrated approach allows linguistic information to play a role in 'planning', with the sequential approach planning takes place without reference to linguistic realization. The third approach, the **interleaved approach**, likewise treats planning and realization as separate processes,

[15]Critics monitor plan interactions and suggest modifications. Later versions of KAMP used TELEGRAM (TELEological GRAMmer) which at least demonstrated Appelt's commitment to modularization of linguistic knowledge at some level.

or modules, but this time there is a *bi-directional* dataflow: i.e. the planning process exports a plan to the realization process and this latter process can request further information from planning process. This approach is manifest in PAULINE (cf. [**?**]) and POPEL (cf. [**?**]). In these systems, the planning process exports a *partial* plan to the realization module and this module attempts to generate the appropriate linguistic representation. The realization process exports requests to the planner when the realizer is unable to generate a linguistic utterance because there is more than one possible realization (or none). This is the case, for example, with referential objects which can be realized as pronouns or definite referring expressions. With the interleaved approach, then, planning can be partially guided by the realization process.

Of these approaches, the interleaved approach seems least problematic and most appropriate for the Sundial system. However, we have extended current interleaved approaches by making a clear distinction between two types of planning, namely **global planning** and **local planning**. Global planning deals with non-linguistic planning: that is, on the basis of information passed from processes within the dialogue manager, it creates a global plan for the message. This plan is partial in the sense that it can be further extended at a later stage. Non-exhaustive global planning is not only plausible from the standpoint of cognitive processing (cf.[**?**]), but facilates real-time message generation since a rough plan can be quickly created and immediately passed to the realization process. Local planning deals with linguistic planning: i.e. it makes decisions as to the specific semantic description of entities and relations on the basis of the current plan as well as other information available within the dialogue manager. Putting this functionality in the MP allows the realization process to remain comparatively simple and compatible with the generation grammar.

However, one consequence of partitioning planning in this way is that the interaction between planning and realization becomes more complex. The global planning process produces a partial plan which is passed to the realization process. The realization process makes requests to, and so interacts with, the local planning process. In deciding on the appropriate answer the local planning process has access to the current global plan as well as other dynamic information in the dialogue manager. In answering these requests not only does the local planning process inform the realization process of its decision, but it also informs the global planning process. This permits the plan to be modified on the basis of actual realization needs even to the extent of global re-planning the message[16]. The interaction between these processes is shown in figure **??**.

One of the advantages of this approach is that it facilitates incremental generation[17].

[16]Whether re-planning will be permitted in the Sundial system is still an open issue.
[17]This applies to the interleaved approach in general.

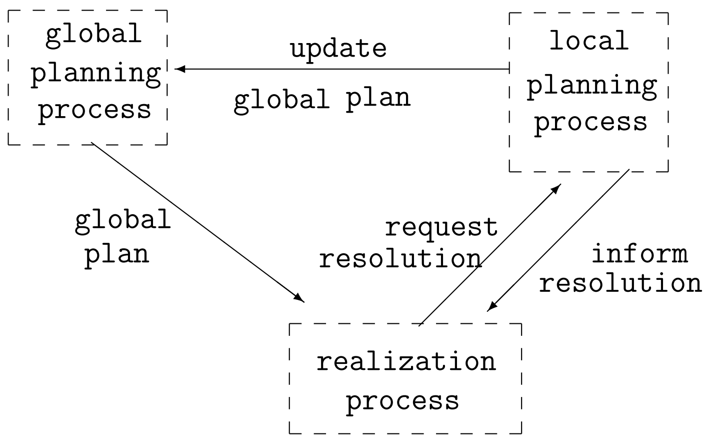

Figure 6.2: Interaction between global planning, local planning and realization processes

Incremental generation occurs when a generation process can inform another process of some of its result before it has completed its work on the message; for example, the planning process can plan some of the utterance and pass this part of the plan onto the realization process and carry on planning the rest of the utterance. It can be contrasted with batch generation where each process informs other processes only when it has completed its work. Thus, given that realization is split into a number of processes corresponding to levels of linguistic description, incremental generation would occur if processes can inform other processes of their results prior to completing all their work. With batch generation, however, all of the message at each level would have to be planned before it would be passed onto the next processing level.

Incremental generation is attractive in many ways. It seems to provide a better model of how people generate language in so far as people start speaking before they know exactly what they want to say in the rest of their utterance (cf. [?]). Additionally, incremental generation would speed up generation since it would permit quick verbalization of part of the message. This could compensate, at least in part, for the tardiness of the speech recognition and synthesis. The problem with incremental generation is that it will inevitably result in false starts and hesitations and this will necessitate the development of (self-initiated) repair strategies (cf. [?] and for a description of these with Conversation Analysis [?]). While these will give dialogic interaction with the Sundial system a highly naturalistic feel, they will, unfortunately, introduce a measure of complexity in the system which is probably beyond our current capabilities. A more attainable goal is *partially incremental generation* where a filter is used to control the output (cf. [?]). The output can be filtered by requiring that utterances are syntactically complete before their morphology and linear order is specified. With this constraint the systems utterances are chunked. Without it, the system would produce utterances word by word.

Adopting this approach to generation requires that what has traditionally been seen as a single process—message planning—be partitioned into two processes, one which globally plans the message and another which locally plans the message. Both these processes are located within the MP in the dialogue manager. Furthermore, the resources and functions of these processes differ from many contemporary approaches to planning since they are components in an *interactive task-driven speech-based* system. This affects the nature of the planning (and realization) process. Firstly, planning must take heed of the dialogic context: the dialogic function of the system message must be 'natural' in light of the current state of the dialogue. It must also allow for the possibility that the system is 'interrupted' and that a planned message may no longer be the most relevant response to the user's utterance. Secondly, the primary function of the system is to give information to members of the public within a limited task domain. One effect of this is that we place greater emphasis on the delivery of the relevant task information than on delivering it in a rhetorical style which matches the particular user[18]. And thirdly, generation is concerned with the planning and generation of voice output, just as interpretation is concerned with the interpretation of voice input to the system. This affects not only the realization of system messages, but also their planning; for example both what is said and how it is said can be influenced by the acoustic reliablity of the user's previous utterance.

The major objective of global planning is to create a schematic structure, or plan, which initiates the linguistic realization process. This structure acts over, and gives shape to, long ranges of the message by specifying dialogic and semantic information. Neither of these types of information are seen as linguistic information: i.e. at this stage in generation, there are no constraints on how this information might be ground out in linguistic representations. Accordingly, the global planning process does not have access to grammatical information. Of course, since the realization process generates the linguistic structure of the message in light of the dialogic and semantic information specified in the plan, there will be linguistic consequences for choices made at this stage in generation. Finally, global planning in this approach is a type of *limited commitment planning* (cf. [**?**]): global planning commits itself to some dialogic and semantic decisions, enough to activate the realizer, but does not commit itself to an exhaustive dialogic and semantic plan, relying instead upon the realization process to call local planning as and when further planning is required.

[18]Both the content and style of system messages have been significantly influenced by the analysis of human-human dialogues and simulations in these task domains.

The global planning process is initiated by a message passed from the input/output decider in the dialogue manager. This message specifies dialogic and semantic information to be included in the next system messsage. The planning process extends this information in two phases. In the first phase, it performs *dialogic planning*: it decides how this information is organized[19]. This results in a message which consists of one or more sequentially ordered components, or 'chunks', each of which are associated with a dialogue act and semantic information[20]. In the second phase, it performs *semantic planning*: it takes the first component and plans its semantic structure in more detail deciding, for example, on the priority of different entities and eventualities as well as their parameters. At this point, the first component is passed to the realization process and the global planning process starts planning the semantic content of the second component. This resulting global plan may be updated with information supplied by the local planning process.

**When to generate** The decision to generate is taken by the input/output decider: a system message is generated when the decider sends a message to global planning. This message is a stream of goals which are ordered in terms of the priority; i.e. the first goal in the stream has the highest priority. These goals have been sent to the decider via 'inform decider' ports on each of the dialogue manager modules. The priority ordering is determined incrementally by an ordering process in the decider.

The function of these goals, in terms of obligations on the system to speak, is reflected in the dialogic and semantic information they contain. For example

- **task goals** provide task information requested by the user; seek confirmation of a user's request; seek confirmation of an inference or default made by the system about the request; seek clarification of a user's request; and repair the user's beliefs about the task or some task parameter.
- **linguistic goals** seek clarification of (part of) the user's utterance; i.e. some word or words have not been understood.
- **dialogic goals** manage the dialogue; for example, openings, closings, 'hold' instructions, redirection, 'satisfaction' checks.

**Dialogic Planning** There are various approaches we could adopt to dialogic planning.

[19]We re-iterate: structuring the goal may be performed by the input/output decider itself (see chapter 5).

[20]Semantic content is intended to cover representations in belief and task modules.

- the approach in TEXT (cf. [?]) where just one of the goals is selected from the stream.
- the approach of [?], [?] and [?] which expands an initial goal, optimizes its structure and then expands the second goal and so on.
- the approach in POPEL (cf. [?]) where the first goal in the stream is a 'seed' for building the system's message.

The approach taken in Sundial synthesizes the second and third approaches. The reason for this is that while it is desirable to plan and realize goals as they appear in the message stream, so respecting the relative priority of goals, it is just as important that the system's messages are 'expanded' and 'optimized' so as to incorporate additional relevant information in the same message plan. Not only can the dialogue planning process merge incoming goals, it can also expand them so as to add information facilitating the user's understanding of the message. Accordingly, the global planning process plans a message to realize all the goals in the stream, but does so incrementally. The first goal is a 'seed' for the system message. The process check to see if this goal needs to be expanded. If it does, then one or more message components, or 'chunks', are generated. Then the process checks to see if the goal can be merged with its successor goal. If the successor goal cannot be incorporated into the plan for the first component, then it becomes the 'seed' for a second component and the process checks to see if it needs to be expanded or can be merged with its successor goal. This continues until no unplanned goals remain in the stream. Dialogic planning then is a breadth first process which outputs a sequence of message components whose order is reflected in the final generated utterance.

The dialogic planning process operates on the basis of sets of dialogue planning rules. One set of rules, expansion rules, test whether the current goal needs to be expanded and if it does, specifies the number, contents and order of the message components. Another set of rules, merging rules, test whether a goal can be merged with its successor goal. The result of applying these rules is a system message composed of sequentially ordered components: the components specify the dialogic and semantic content of the message; and the sequential relationship between the components affects the pause, or 'gap', between the realizations of these components. Thes rules operate on the basis of the following information

1. dialogue act information
2. referential information
3. task information

4. linguistic information, including, where necessary, the confirmation level of the user's linguistic expression of the relevant semantic parts. Confirmation levels are calculated on the basis of the acoustic scores of the user's utterance in which the information was delivered.

5. the satisfaction status of the current goal.

The operation of dialogue planning can be seen from the treatment of the following examples of system messages which might occur within the request confirmation phase of a flight information dialogue (see chapter 2 of WP7, D1 for further details).

(11) a U: I want to check the arrival time of the BA 285 from Paris
S: **the arrival time of flight number BA 285. Is that correct?**

b U: I want to check the arrival time of the BA 285 from Paris
S: **flight number BA 285 leaving from Paris**

c U: I want to check the arrival time of the BA 285 from Paris
S: **BA 285 from Paris. When does it leave?**

d U: I want to check the arrival time of the BA 285 from Paris
S: **okay. When does it leave?**

What causes these differing responses from the system is the acoustic reliability of the user's request: they range from dubious relibility in (11a) through to excellent reliablity in (11d). These differences in reliability affect the behaviour of the dialogue manager in the following way. With (11a), the system has understood that an arrival time has been requested, the airline is BA and the flight number is 285; it has not understood the departure location. The understood request parameters have a low acoustic score. Since the acoustic score is so low, the representation of the flight in the belief model is not sufficiently reliable for the task model to move onto the next task (checking that the flight is sufficiently well specified for database access). Accordingly, the task module sends three confirmation goals to the input/output decider where each goal includes a specification of the confirmation level. These goals are then given a priority ordering. As far as we can see the order is irrelevant here. This goal stream is then passed to global planning. Both expansion and merging rules apply with these goals since the confirmation level is very low. A merging rule 'packs' the three goals into a single message component and an expansion rule introduces a second component whose function is to check that the information is correct. These components are ordered so that there will only be a short pause between them.

The remaining examples in (11) are planned in an analogous manner. With (11b), all parameters are understood by the system (i.e. they are present in the belief model). They are assigned a confirmation level which results in the task model passing a set of confirmation goals to the decider. In the dialogic planning phase, while no expansion rules apply (the confirmation level is sufficiently low), a merging rule packs these goals into a single component. With (11c), the confirmation level is higher and, in addition to a set of confirmation goals, a clarification goal is also sent to the decider. For this level of confirmation permits the task module to move onto the next task and, in this case, it is requesting a misssing parameter[21]. These goals are ordered by the decider so that confirmation goals precede clarification goals. Again, only merging rules apply. With the fourth system message, (11d), the acoustic scores on the parameters merit a high confirmation level. This permits the task module to move onto the next task and send out a clarification goal. The 'weak' confirmation, *okay*, does not come about from a goal sent by the task module: the confirmation level is sufficiently high for the task module to proceed without confirmation. However, it seems appropriate that the system lets the user know that their request has been understood. Accordingly, a phatic confirmation goal can be sent out by the dialogue module[22].

Dialogic planning can also be illustrated with system messages generated in the request resolution phase of flight information dialogues. The goals originating from the task module in this phase depend upon not just the original request from the user, but also the response from the database, especially in cases where there is a successful, but mis-matching, database call, an unsucessful database call and a successful, but vague, database call.

With sucessful but mis-matching database calls, the mismatching arises from constraint relaxation: one or more parameters have been relaxed so as to obtain a database response. This creates a discrepancy between the representation of the flight in the user believes partition of belief model and the representation given by the database. In order to create a successor flight representation, i.e. a representation of the flight information which will be put in the 'mutual belief' partition of the belief model, the task module needs to **replace** one or more of the parameters in the original flight representation given by the user. Accordingly, not only does this successor flight representation **add** information to the original flight representation, but it needs to replace one or more parameter values with values obtained from the database representation. In sending a goal to the decider these substitutions need to be marked so that MG can generate messages with the appropriate prosodic markings such as emphatic prominence. One possibility

[21]To clarify how the task module operates: with low confirmations, a successor task is not calculated and only confirmation goals are sent to the decider; with higher confirmation levels, both confirmation and clarification goals can be sent out.

[22]It is unclear at present whether such system utterances should be generated at all: users may prefer the reassurance of having all information echoed by the system.

is that values with substituted parameters are marked as **emphasis** parameters.

With unsucessful database calls, the goal from the task module needs to indicate that the requested flight does not exist[23]. In this case, the successor flight representation needs to include all the information specified by the user, but also needs to note that this is an inconsistent state of affairs: there is no corresponding flight represented in the database and thus the representation is inconsistent from the system point of view. This can be represented with a successor flight representation where the MODE of the flight entity is *no*.

Successful but vague requests cause the database to return more than one flight representation. This is represented as a complex entity in the task model, where complex entities are sets of individuals, here flights. Each member of the set describes one of the flights matching, or perhaps mis-matching, the one requested by the user.

In each of these cases, the task model sends a single goal to the decider. The behaviour of the dialogic planning phase of global planning varies with the structure of these goals. Where the goal contains an single individual flight with no emphasised parameters, dialogic planning simply creates a single message component. Where the database call has been unsucessful, i.e. the object in the goal has MODE *no*, the expansion rules create more than one component. The first component of the structure includes the information in the goal: i.e. that there are no flights matching the user's description. This is eventually realized as *there are no* user's flight description. The second component, which is separated from the first by a small gap, requests further information from the user; for example, *do you want information about another flight?*. The last case, where there are multiple (mis-)matching flights, always requires goal expansion. The first component is identical to that for an unsuccessful database call, in that it asserts the existence of a number of flights, where the number corresponds to the number of members in the set. This first component is eventually realized as *there are* NUMBER user's flight description'. How the goal is further expanded depends upon whether there is a simple regularity in the flights as well as their number. Simple regularities concern, for example, simple patterns in the arrival and departure times – they may all leave at hourly intervals. If there is a simple regularity, then this is expressed in the second component of the message, giving rise to utterance chunks such as *All flights leave Paris on the hour* or *they leave Paris at nine and ten am respectively.* A third component is then introduced, with a longer gap, which requests further guidance from the user; for example, *do you want further information about one of these flights or some other flight?* If there is no simple regularity, then the behaviour of the dialogic phase depends upon

[23]We assume that there will be few instances of this since constraint relaxation processes will typically suggest another 'similar' flight.

their number. If the number is less than or equal to some arbitrary number – say three – then a separate message component is planned to give the information for each flight. In this circumstance, the first component informs the user that there are a number of flights and a set of subsequent components gives the requested information for each of these flights. Another component can then be added, with a long pause, which request further guidance from the user; for example, *do you want further information about one of these flights?* Finally, if there is no simple regularity in the flights and there are more than three, then the second component is a request for further guidance, perhaps *Can you give me further information about the flight?*

In sum, dialogic planning is a breadth-first planning process which expands and optimizes goals from the decider. This results in a message structure which consists of one or more message components ordered in terms of short or long gaps. This message structure, the global plan, is then passed to the semantic planning phase of the global process.

**Semantic Planning** Semantic planning is a depth-first process: the first component is planned (and passed to the realization process) before any planning on the second takes place. Semantic planning creates a predicate argument structure for each component which serves two functions[24]. The first is to provide a representation of the message component which the realization process is able to start work on[25]. The second is to structure this information so that the resulting message component is thematically coherent from the user's point of view.

Semantic planning performs three tasks:

The first task is to create a predicate argument structure for the component. This involves determining how many predicate-argument elements are required and what information should go in each one. That is, it decides the relationship between semantic content and predicate-argument structure for a component. In the case of requesting further information about a user's flight, for example, the structuring rules will create two arguments and a predicate: one argument specifies the flight and another the value of the parameter that is being sought. The predicate is not itself specified but since its arguments are, this imposes constraints on which lexical entries will match the predicate.

[24]This structure parallels the Function Semantic Structure in POPEL.

[25]Recall that global planning has no linguistic knowledge. Thus it does not produce a structure that contains syntactic information. The semantic structure it produces is 'non-linguistically' oriented in the sense that it does not take into account the semantics of individual lexical entries. We assume that the realization processes contain rules which link 'non-linguistically' orientated semantics to lexical and phrasal semantics – i.e. concept to sign rules.

Its second task is to determine the structure and relative priority of parameter-value pairs in each of the arguments and predicates. Determining the structure can involve the optimization of parameter value pairs where two (or more) pairs are conjoined into a composite parameter. This is the case, for example, with company and flight number parameters in the flight domain which are packed into a composite parameter 'flight identification'. These parameter-value pairs are then given a priority ordering.

The third task is to make ordering decisions regarding the elements of the message, which may be realised in the *thematic organisation* of the clause produced. There are two basic ways in which such ordering information could be encoded in the global plan:

1. by encoding such information as features on the elements of the message. This approach is typically used in systemic generation;
2. by outputting the elements of the message in a stream which is ordered, or partially ordered, in a way that reflects the desired thematic organisation.

The second solution has more explanatory power, if the ordering of elements can be derived according to consistent criteria, such as relative newness, importance, etc, and if such an ordering easily mapped onto one of the surface stylistic choices allowed by the language. On the other hand, the degree to which relative importance/newness is encoded in thematic organisation, as opposed to, say, prosodic emphasis, may be language dependent. In English, I can say:

(12) I haven't seen JOHN

—with a fall-rise contour on "john" implying, perhaps, that I have seen somebody else. Italian on the other hand has less flexibility regarding the use of prosodic accent to single out elements, and uses word-order variation instead:

(13) Giovanni non ho visto

Such 'topicalisation' would be stylistically awkward in English.

Further work is needed to determine to what extent pragmatic properties of an utterance may be encoded by ordering its elements, and to what extent they need to be encoded featurally. Prosodically realised properties, for example, cannot be encoded by an ordering of elements. It may be that a homogenous approach will encode all pragmatic properties featurally. If this is to be the case, further work will be needed to determine a proper logic governing the semantics and cooccurrence of these features.

**Local Planning** The local planning process plans elements in the message component of the global plan. Planning is local in the sense that the process only plans the argument or predicate which the realization process is currently attempting to realize linguistically. It can therefore be described as **formative** planning (cf. [**?**]): it controls the content and structure of part of an utterance.

The aim of local planning is to guide linguistic generation so that the global plan is **semantically complete**: i.e. each argument and predicate in a messsage component is lexically ground. An argument or predicate is lexically ground when it is part of a unique sign. This aim is achieved by means of local semantic planning in response to requests sent by the realization process. The local planning process replies to requests from the realization process to make semantic decisions. Semantic decisions concern the exact semantic description of arguments and predicates in the message component.

There are three other noteworthy properties of local planning.

- **opportunistic** Local planning can result in the introduction of predicate-argument structure which was not part of the initial global plan. Further structure is introduced as a consequence of the syntactic requirements of signs: e.g. a verb requires an argument which was not planned for at the global level. So, while local planning generally plans the semantic descriptions of existing elements of predicate-argument structure, it can, if necessary plan the description of further elements if this is required in order to produce syntactically well-formed utterances. Of course, when this occurs the global plan will need to be updated.

- **re-planning** There may be case where linguistic realization process is unable to realize the global plan and further local planning fails to resolve this difficulty. In such cases the local planning process receives a 'fail' message from the realization process and it, in turn, informs global planning that a message component requires replanning. Whether this is desirable (or necessary) in the Sundial system is debatable.

- **global information** In addition to information from the global planning and linguistic realization processes, local planning also draws upon information from modules within the dialogue manager module in order to make local semantic and syntactic decisions (not any other modules though). In particular, it has access to the belief model and focus processes in the belief module, the task model in the task module, the linguistic history in the linguistic processing interface and the dialogue history in the dialogue module.

**When to begin local planning** Local planning begins in response to a request from the linguistic realization process. Local planning ends when the message component is semantically complete. That is, when all elements in the predicate-argument structure have been lexically ground, all subsequent requests from the realization process are refused. When its requests are refused, the realization process immediately attempts to determine the morphology and linearization of the resultant sign.

**Deciding semantic descriptions** The task of the local planning process is to *extend* the semantic description of arguments and predicates so as to enable the realization process to match the result against a lexical or phrasal entry using concept-to-sign rules. How this is done depends on whether the element in question is given or new. Given elements are those with representations in the belief model; new elements are those without a corresponding representation. Here we illustrate how local planning makes semantic decisions about arguments rather than predicates.

Arguments can be realized in two orthogonal ways: a recent linguistic description (from the linguistic history) can be *re-used* or a new description can be *created.* There are various methods by which a new description can be created.

1. By using an anaphoric pronouns such as *he, she, it, they* or deictic expressions such as *the first, second* or *last.*
2. By using a description which has minimum referential adequacy: the description is sufficient to distinguish it from entities in local focus, but may be ambiguous within the belief model.
3. By using a description which has maximum referential adequacy: the description is sufficient to distinguish it from all entities in the belief model.

With given arguments, local planning must decide whether or not new information is being predicated of them, and if so include such predications in its description. Otherwise it may modify a recent expression from the linguistic history, use a pronoun, or use a mimimum or maximum expression, depending on the extent to which the entity is in focus. The assigment of prosodic features reflecting the degree of focus is along similar lines; in English there are two basic ways of encoding ‘givenness’ prosodically, either by deaccenting or by the use of a fall-rise, or ‘referring’ accent. The decisions regarding which type of focus is used are to some extent orthogonal to those regarding choices of referring expression; broadly speaking, they can be said to based on some assessment of relative ‘accessibility’ by the hearer.

New entities ususually require full descriptions of their properties, or sufficient description to relate them to existing entities. This may also be done by the modification of a recent description in the linguistic history.

### 6.3.2 Information flows between the DM and the MG

While the realization process is part of the message generation module, both local and global planning are part of the message planning module in the DM. There are two principal reasons for this. The first is that both make planning decisions, linguistic or otherwise, which guide how the message is to be realized linguistically. The second reason is that these processes draw upon dynamic information in the dialogue manager; for example, global planning draws upon goal information from the decider and local planning upon referential, task and linguistic information stored in modules of the dialogue manager module. Accordingly, the realization process in message generation module does not need to access any information stored in dialogue manager modules other than the message planning module.

The interface is specified as follows. In all interactions, messages passed to and fro are expressed in the interface language. The MP in the DM sends a global plan to the MG which it attempts to realize linguistically. This plan is a result of the global dialogic and semantic planning. The MG sends requests to the DM for further descriptions of referential expressions. These requests are handled by local planning in the MG which sends the results back to the MG. Furthermore, the MG feeds back the results of the linguistic realization. For, as a result of speaking, the system needs to know both when it has uttered a message as well as the structure and content of that message. Accordingly linguistic information is passed to the LI so the linguistic history can be updated. The MP also feeds back dialogue and semantic decisions to other modules in the DM, in particular, the belief, task and dialogue modules. Feedback is performed synchronously: i.e. the MG passes a linguistic representation of the system utterance to the MP and as a result of this message, it sends out messages which causes updating of the relevant histories and models within the DM. Finally, updating occurs at the end of each message component. This ensures that semantic descriptions are locally planned on the basis of current user and system beliefs (including topic information) and also facilitates global replanning in light of interruptions.

This interface specification is illustrated in figure **??**.

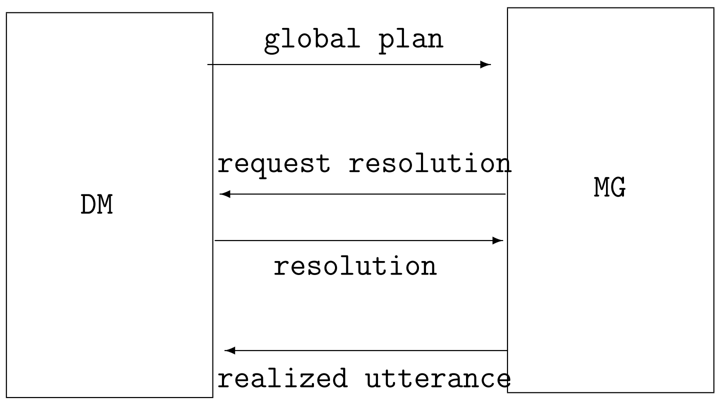

Figure 6.3: The DM–MG Interface

### 6.3.3 Further Issues

1. **canned generation** The description of the MP and the interface is based upon the assumption that linguistic realization will follow a linguistic rules approach (see WP7, D1, chapter 5). Replacing rule-based generation with canned generation (see WP7, D1, chapter 4), however, has only a minimal effect on what has been said so far. Canned generation can be subdivided into *complete* and *parameterized* canned realization.

   With completely canned realization, each utterance is assigned a unique index where the representation lacks 'holes' or parameterized arguments which need to be further instantiated. Generation of utterances takes place in three phases: the construction of a global plan for the utterance; translation of the plan into a unique index; and conversion of the index into a completely specified linguistic representation which drives the speech synthesizer. In order to translate the global plan into a unique index, the global plan must be sufficiently specific to allow the assignment of a single index. With this approach, global planning remains as described above, but local planning is unnecessary. We have yet to resolve whether the process assigning an index to the global plan is part of message planning or linguistic realization. Adopting the latter strategy would result in an DM–MG interface which is identical to that for rule-based generation, again with the exception that local planning is omitted.

   Parameterized canned realization differs from complete realization in so far as there are arguments in the indexed representation which need to be specified before the representation can be passed to the speech synthesizer. These arguments can be further specified via local planning: i.e. the MP is sent a request for further specification of the parameterized arguments just as it is sent requests to instantiate referential expressions. This results in exactly the same processes and interfaces as for rule-based realization.

2. **recalling planned components** We still need to determine how planned components can be recalled from the MG when a user's interruption makes a component unnecessary or inappropriate. One solution is that we extend the interface language so as to include a 'recall component' feature. This feature would have the same status as 'total failure' messages from the linguistic processing module.

3. **top-level syntactic category** Many rule-based generation systems require the top-level input representation to be assigned a syntactic category. Currently, we are exploring whether to place this assignment function in global planning or linguistic realization.

4. **syntactic decisions** There will be various point in the linguistic realization process where syntactic decisions have to made; for example, which subcategorization frame to use for a given verb. Syntactic decisions could be made either by an extended local planning process, partially on the basis of the global plan, or by default rules in linguistic realization. If these decisions, as well as the assignment of a top-level syntactic category, are made by the linguistic realization process, then no linguistic decisions need be made in the MP.

# Bibliography

Allen, J. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM,* 26(11):832–843.

Allen, J. (1989). Recognising intenstions from natural language utterances. In M. Brady and R. Berwick, editors, *Computational models of discourse.* MIT Press.

Allen, J. and P. Hayes (1985). A common-sense theory of time. In *Proceedings of the nineth international joint conference on artificial intelligence,* pages 528–531.

Allen, J. and C. Perrault (1980). Analysing intention in utterances. *Artificial Intelligence,* 15:143–178.

Appelt, D. E. (1982). Planning natural-language utterances. In *2nd AAAI Conference Proceedings, Pittsburgh.*

Appelt, D. E. (1983). Telegram: a grammar formalism for language planning. In *8nd IJCAI Conference Proceedings, Karlsruhe.*

Appelt, D. E. (1985). *Planning English sentences.* Studies in Natural Language Processing. Cambridge University Press.

Barwise, J. and J. Perry (1983). *Situations and Attitudes.* MIT.

Bobrow, D. G., R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson, and T. Winograd (1977). Gus, a frame-driven dialogue system. In *AI Journal.*

Brachman, R., V. Gilbert, and H. Levesque (1985). An essential hybrid reasoning system: knowledge and symbol level accounts of krypton. In *Proceedings of the nineth international joint conference on artificial intelligence,* pages 532–539.

Brietzmann, A. (1984). Semantische und pragmatische analyse im erlanger spracherkennungsprojekt. Master's thesis, Lehrstuhl für Informatik 5 (Mustererkennung), Universität Erlangen–Nürnberg, Erlangen.

Brown, G. and G. Yule (1983). *Discourse Structure.* Cambridge University Press.

Bublitz, W. (1988). Discourse topics and topical actions, participant roles and 'recipient action' in a particular type of everyday conversation. In *Supportive fellow-speakers and cooperative conversations.* Benjamins, Amsterdam and Philadephia.

Bunt, H. (1986). Information dialogues as communicative action in relation to partner modelling and information processing. In *Structure of multimodal dialogues including voice / Structure de dialogues multimodaux à composante orale,* Venaco, France. NATO RSG10/RSG12.

Bunt, H. (1988). Mens-machine dialoogvoering volgens de regels. *Interdisciplinaar Tijdschrift voor Taal- en Tekstwetenschap,* 8(2):193–208.

Bunt, H. (1989). Information dialogues as communicative action in relation to partner
modelling and information processing. In M. M. Taylor, F. Néel, and D. G. Bouwhuis, editors, *The structure of multimodal dialogue,* pages 47–71. North-Holland.

Bunt, H., J. Thesingh, and K. van der Sloot (1985). The tendum dialogue system and its theoretical basis. *IPO Annual Progress Report,* 19:105–113.

Carberry, S. (1985). A pragmatics-based approach to understanding intersentential ellipsis. In *Proceedings of the Association for Computational Linguistics,* pages 188–197.

Carberry, S. (1988). A pragmatics-based approach to ellipsis resolution. *Computational Linguistics,* 15(2):75–96.

Cutler, A. and D. Ladd (1983). *Prosody: Models and Measurements.* Springer Verlag.

D. Delomier, A. Meunier, M. M. (1989). Linguistic features of human-machine oral dialogue. In *Eurospeech 89,* Paris. Eurospeech Congres.

Dale, R. (1989). Generating recipes: An overview of epicure. Human Communication Research Centre, University of Edinburgh.

Danlos, L. (1987). *The linguistic basis of text generation.* Studies in Natural Language Processing. Cambridge University Press.

Desmedt, K. and G. Kempen (1987). Incremental sentence production, self-correction and coordination. In G. Kempen, editor, *Natural Language Generation.* Dordrecht, Martinus Nijhoff.

Ehlich, K. and J. Rehbein (1976). Halbinterpretative arbeitstranskriptionen (hiat). *Linguistische Berichte,* 45:21–41.

Ehlich, K. and J. Rehbein (1979). Erweiterte halbinterpretative arbeitstranskriptionen (hiat 2). *Linguistische Berichte,* 59:51–74.

Falzon, P. (198?). Communication homme-homme et interaction homme-machine. Bulletin de recherche de l'inria, INRIA, Rocquencourt.

Fenstad, J. E., P. Halvorsen, and T. Langholm (1987). *Situations, Language and Logic.* Number 34 in Studies in Linguistics and Philosophy. D. Reidel.

Finkler, W. and G. Neumann (1989). Popel-how a distributed parallel model for incremental natural language production with feedback. In *11th IJCAI Conference Proceedings, Detroit.*

Garrod, S. and A. Anderson (1987). Saying what you mean in dialogue: A study in conceptual and semantic co-ordination. *Cognition,* 27:181–218.

Gilbert, G. N., R. Wooffitt, and N. Fraser (1990). Organising computer talk. In P. Luff, N. Gilbert, and D. Frohlich, editors, *Computers and Conversation.* Academic Press.

Good, D. (1986). The viability of conversational grammars. In *Structure of multimodal dialogues including voice / Structure de dialogues multimodaux à composante orale,* Venaco, France. NATO RSG10/RSG12.

Grice, H. P. (1975). Logic and conversation. In P. Cole and L. Morgan, editors, *Syntax and Semantic, vol III, Speech acts,* pages 41–58. Academic Press.

Grosz, B. (1977). The representation and use of focus in dialogue understanding.

Grosz, B. (1981). Focusing and description in natural language dialogues. In B. W. Joshi, A.K. and I. Sag, editors, *Elements of discourse understanding.* Cambridge University Press.

Grosz, B. J. and C. L. Sidner (1985). Discourse structure and the proper treatment of interruptions. In *Proceedings of the IJCAI,* pages 832–839.

Grosz, B. J. and C. L. Sidner (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics,* 12(3):175–204.

Grosz, B., A. Joshi, and S. Weinstein (1983). Providing a unified account of definite noun phrases in discourse. In *Proceedings of the Association for Computational Linguistics,* pages 44–50.

Guyomard, M. and J. Siroux (1989). Les actes de dialogue. Technical report, IRISA, Lannion.

Hendrix, G. G. (1979). Encoding knowledge in partitioned networks. In N. V. Findler, editor, *Associative Networks,* pages 51–92. Academic Press.

Heritage, J. (1984). *Garfinkel and Ethnomethodology.* Polity Press: Cambridge.

Hirschberg, J. and J. Pierrehumbert (1986). The intonational structuring of discourse.

Hitzenberger, L. e. a. (1986). Facid. fachsprachlicher corpus informationsabfragender dialoge. Technical report, Linguistische Informationswissenschaft Universität Regensburg, Regensburg.

Houghton, G. (1986). *The production of language in dialogue—a computational model.* PhD thesis, University of Sussex.

Hovy, E. H. (1988). *Generating Natural Language Under Pragmatic Constraints.* Lawrence Erlbaum Associates.

Jefferson, G. (1983). Notes on some orderliness of overlap onsets. *Tilberg papers in Language and Literature.*

Kaplan, J. (1983). Cooperative responses from a portable natural language database query system. In M. Brady and R. Berwick, editors, *Computational models of discourse.* MIT Press.

Kelleher, G. and B. Smith (1988). A brief introduction to reason maintenance systems. pages 4–20. Ellis Horwood.

Kowalski, R. and M. Sergot (1986). A logic-based calculus of events. *New Generation Computing,* 4:67–95.

Levinson, S. C. (1983a). *Pragmatics.* Cambridge Textbooks in linguistics.

Levinson, S. C. (1983b). *Pragmatics.* Cambridge Textbooks in Linguistics. Cambridge University Press.

Luzzati, D. (1986). Alors: un analyseur sélectif adapté au traitement d'énoncés oraux spontanés. Technical report, LIMSI.

McDonald, D. D. (1985). Surface generation for a variety of applications. In *National Computing Conference.*

McDonald, D. D. and J. D. Pustejovsky (1985). Description-directed natural language generation. In *IJCAI Conference Proceedings, Los Angeles.*

McDowell (1989). Knowledge representation for commonsense reasoning with text. *Computational Linguistics,* 15.

McKeown, K. R. (1985). *Text generation using discourse strategies and focus constraints to generate natural language text.* Studies in Natural Language Processing. Cambridge University Press.

McKeown, K. R. and W. R. Swartout (1988). Language generation and explanation. In M. Zock and G. Sabah, editors, *Advances in Natural Language Generation.* Pinter Publisher, London.

Mellish, C. (1988). Natural language generation from plans. In M. Zock and G. Sabah, editors, *Advances in Natural Language Generation.* Pinter Publisher, London.

Mellish, C. and R. Evans (1989). Natural language generation from plans. *Computational Linguistics,* 15(4).

Moeschler, J. (1986). *Pragmatique conversationnelle: aspects théoriques, descriptifs et didactiques.*

P.Cohen (1978). *On knowing what to say: planning speech acts.* PhD thesis, University of Toronto.

Pollard, C. and I. A. Sag (1987). *Information-based syntax and semantics.* Number 13 in CSLI Lecture Notes. Centre for the Study of Language and Information.

Ponamale, M., A. Lucas, B. Pelletti, S. Pavan, A. Cozannet, N. Fraser, and K. Choukri (1990). Simulations and corpora analysis. Technical Report WP3 Deliverable No.1, Cap Gemini Innovation.

Power, R. (1979). The organisation of purposeful dialogues. *Linguistics,* 17:107–152.

R. Amalberti, N. Carbonell, P. F. (1984). Stratégies de contrôle du dialogue en situation d'interrogation téléphonique. In *Communication orale homme-machine.* GALF-GRECO.

Reichenbach (1947). *Elements of symbolic logic.* Macmillan.

Reichmann, R. (1985). *Getting computers to talk like you and me.* MIT press.

Reiter, R. (1980). A logic for default reasoning. *Artifical Intelligence,* 13.

Reithinger, N. (1989). Popel - a parallel and incremental natural language generation system. presented at the 4th International Workshop on Natural Language Generation, Santa Catalina, CA 18-20th July 1988.

Richards, M. and K. Underwood (1984). How should people and computers speak to each other? In *Interact,* pages 33–36.

Roulet, E. (1986). *Complétude interactive et mouvements discursifs.*

Schlegoff, E. A. and H. Sacks (1973). Opening up closings. *Semiotica,* 7(4):289–327.

Schriefers, H. and T. Pechmann (1988). Incremental production of referential noun phrases by human speakers. In M. Zock and G. Sabah, editors, *Advances in Natural Language Generation.* Pinter Publisher, London.

Sidner, C. (1983). Focusing in the comprehension of definite anaphora. In M. Brady and R. Berwick, editors, *Computational models of discourse.* MIT Press.

Siroux, J. (1986). Pragmatics in the realisation of a dialogue module. In *Structure of multimodal dialogues including voice,* Venaco (france). NATO.

Sowa, F. (1984). *Conceptual Structures: Information processing in Mind and Machine.* Addison Wesley.

Thompson, H. (1977). Strategy and tactics: a model for language production. In *Papers from the 13th Regional Meeting, Chicago Linguistic Society.*

Vilnat, A. (1984). *Elaboration d'interventions pertinentes dans une conversation homme–machine.* PhD thesis.

Wachtel, T. (1986). Pragmatic sensitivity in nl interfaces and the structure of conversation. In *Proceedings of the 11th international conference on Computational Linguistics,* pages 35–39.

Wahlster, W. and A. Kobsa (1985). Dialog-based user models. Technical report, XTRA Report (University of Saarbrucken).

Ward, G. and J. Hirschbirg (1985). Implicating uncertainty: the pragmatics of fall-rise intonation. *Language.*

Wilensky, R. (1983). *Planning and understanding: a computational approach to human reasoning.* Addison-Wesley.

Winograd, T. (1972). *Understanding Natural Language.* Academic Press.

Woods, W. A. (1985). Language processing for speech understanding. In W. W. F. Fallside, editor, *Computer Speech Processing,* pages 305–334. Prentice Hall.

Woods, W. (1986). Important issues in knowledge representation. *Proceedings of the IEEE,* 74(10):1322–1334.

Young, S. and C. Proctor (1989). The design and implementation of dialogue control in voice operated database enquiry systems. *Computer Speech and Language,* 3(4):329–354.