Classification of Public Transport Information Dialogues using an Information Based Coding Scheme

R.J. van Vark[1,2] and **J.P.M. de Vreught**[1,2] and **L.J.M. Rothkrantz**[1,2]

Abstract. The goal of a recently started research project of OVR is to develop a system to automate part of its dialogues now held by human operators at its call centres. To achieve a well thought-out design of such a system both an analysis of the current dialogues and an experiment determining which form of dialogues would be favoured by humans were started. In this paper we will emphasize the analysis.

Over 5000 dialogues were recorded and transliterated. Our coding scheme based on the information components has been derived from this set. The coding scheme was applied to a sample of nearly 500 dialogues. Not only are we able to describe the semantics of the utterances in a dialogue, but we are also able to classify the dialogues with respect to their difficulty for an automated speech processing system.

With a perfect speech recogniser and parser, in theory 88% of the dialogues could be handled by a simple system that would work with question forms. This is not realistic since speech recognition is a serious bottleneck. Dialogues about train information only are preferable over dialogues about other transport modalities for such a system, since these dialogues are shorter and less complex.

1 INTRODUCTION

OVR, a Dutch provider of information concerning national public transport, has recently started an investigation of the usability of Automated Speech Processing (ASP) in their company. In the future, part of their humanly operated information service would have to be automated by using ASP. OVR is involved in European projects on the retrieval of information concerning public transport like MAIS, RAILTEL, and ARISE.

Besides developing a demonstrator for a subset of its information dialogues, the current research project covers the examination of a large corpus of over 5000 dialogues recorded at its call centers. Examining such a large corpus serves a number of purposes:

- developing a standard for characterising information dialogues and services,
- designing and implementing a dialogue manager to manage information transactions,
- examining user expectation of such information services and automated versions thereof,
- identifying the inherent structures and statistical relationships of dialogues by empirical methods.

This article is aimed at describing an effective coding scheme to study information dialogues and to describe how difficulty of information dialogues can be measured based on an unambiguous coding scheme and a large corpus.

Related work is discussed first, followed by a description of the dialogues examined. In the fourth section, the coding scheme based on these information dialogues is discussed. Results of applying this coding scheme to OVR dialogues will be presented in section 5, followed by a discussion of automated processing of these dialogues. Finally future work will be described.

2 RELATED WORK

The basic assumption of plan based theory of dialogue modeling [4] is that the linguistic behaviour of agents in information dialogues is goal directed. To reach a particular state agents use a plan which is often a small variation of a standard scenario. The structure of a plan resembles the dialogue structure. Dialogue acts in OVR conversations are also assumed to be part of a plan and the listener is assumed to respond appropiately to this plan and not only to isolated utterances.

Carletta et al. [3] developed a coding scheme for task oriented dialogues existing of three levels. At the highest level dialogues are divided in transactions or subdialogues corresponding to major steps in the participants' plan for completing the task. At the intermediate level conversational games consist of initiations and responses. A conversational game is in fact a sequence of utterances, starting with an initiation and ending with the requested information being transferred. At the lowest level conversational moves are utterances which are initiations and responses named according to their purpose. Our coding scheme has been strongly influenced by their scheme, however, our scheme is both more generic and more detailed.

In the VERBMOBIL project [1] researchers developed a taxonomy of dialogue acts. To model the task oriented dialogues in a large corpus they assumed that such dialogues can be modelled by means of a limited but open set of dialogue acts. They defined 17 dialogue acts and semi-formal rules to assign them to utterances. These assignment rules served as a starting point for the automatic determination of dialogue acts within the semantic evaluation system. From the analysis of an annotated corpus of 200 dialogues they derived a standard model of admissible dialogue act sequences. We also

[1]This research was funded by Openbaar Vervoer Reisinformatie (OVR) and SENTER.

[2]Delft University of Technology, Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, The Netherlands, `ovr@kgs.twi.tudelft.nl`

investigated phase sequences in a similar way (see section 5).

Another approach of modeling dialogues [2, 10] is based on the observation that in goal directed dialogues, the information is exchanged in such a way that the dialogue contains one or more topical chains. These are sequences of utterances that all communicate information about the same topic (topic packets). Successive topics are usually related to each other. Topic transitions in a dialogue are modelled as movements across these topic packets. The local relationship between topics can be used in knowledge based systems in a dialogue manager for ASP systems.

Since the introduction of plan theory researchers criticized a plan based approach. For example Dahlbäck [5] noticed that in advisory dialogues it is possible to infer the non-linguistic intentions behind a specific utterance from knowledge of the general task. In an information retrieval task this inference can be very difficult, however, in order to provide helpful answers this information is not needed frequently. Since the OVR dialogues are strongly goal directed intentions are not very important. In a recently developed demonstrator for the German railways, intentions are deliberately neglicted and this system is strongly goal directed as well [9].

3 DIALOGUES

The coding scheme described in this paper, is developed using a set containing 5205 dialogues [11]. These dialogues consist of telephone conversations between two human agents. One of these agents is an operator working at a call center, while the other agent is a client calling the call center to gain information.

The context in which these conversations take place is well defined, since all clients want to retrieve information about the services offered by Dutch public transport. The piece of information a certain client is looking for, can range from simple fare information to more complex information like complete travel schedules for two streets anywhere in the Netherlands using all different kinds of transport modalities.

The goal both agents will pursue is quite clear. Clients want to get the information about public transport which they desire and the operator wants to serve the client as good as possible. However, sometimes information is presented which is not of interest to the client or of which the client is already aware of.

Two major tasks can be distinguished in all conversations. The first task is to get both parties to understand which information is desired by the client. The second task consists of the operator presenting the information inquired by the client. The information is normally broken down in several pieces which makes it easier for the client to memorise or write down the information. In most conversations both tasks only occur once or twice. However, these tasks can occur more often.

Recording of the dialogues took place from May 1995 to July 1995. These recordings were made by the Dutch PTT using incoming calls at all call centers of OVR. For privacy reasons the recordings were deprived of all personal information like names and addresses. Also the time spent on hold were deleted from the recordings.

In table 1 the frequence table is given for all conversations. Actually, the dialogues were grouped in five categories: train information, bus/tram information, train & bus/tram information, miscellaneous, and failures [11]. The median of the dialogues containing train information only is approximately $\frac{3}{4}$ minute while from all other categories the median lies around $1\frac{1}{4}$ minute.

Table 1. Dialogue duration

duration (sec.)	amount #	amount %	duration (sec.)	amount #	amount %
0- 10	1	0.0	105-120	283	5.4
10- 20	122	2.3	120-150	294	5.6
20- 30	590	11.3	150-180	185	3.6
30- 40	740	14.2	180-210	118	2.3
40- 50	677	13.0	210-240	69	1.3
50- 60	504	9.7	240-270	55	1.1
60- 75	670	12.9	270-300	29	0.6
75- 90	479	9.2	300-360	30	0.6
90-105	326	6.3	360- ∞	33	0.6

Before dialogues were coded, they were first transliterated: the spoken text was transcribed into written text. Transliteration was necessary to gain better insight into the mechanisms underlying the studied dialogues and eventually to derive a coding scheme based on these information dialogues.

```
2:goedemorgen reisinformatie
  (good morning travel information)
1:goedemorgen [achternaam] kunt u mij zeggen hoe
  laat de[uh] bus van Lochem naar Deventer toe gaat
  (good morning [last name] can you tell me what)
  (time the[uh] bus departs from Lochem to Deventer)
2:hoe laat ongeveer zou u mee willen
  (how late approximately would you like to go)
1:[u:h] rond een uur of negen
  ([uh] about nine o'clock)
2:oké
  (ok)
2:negen uur twee mevrouw buslijn zesenvijftig
  (nine o two madam bus line fifty-six)
1:oké dank u wel
  (ok thank you)
2:tot uw #1 dienst #
  (at your service)
1:#1 ja hoor # #2 da:g #
  (yes fine bye)
2:#2 goedemorgen #
  (good morning)
```

Figure 1. Sample conversation

Some aspects have to be taken into consideration when transliterating dialogues. One of these aspects is of course speaker annotation, which was done by annotating each utterance with a starting marker '1:' for clients and '2:' for operators. The other important aspect is to annotate interruptions and simultaneous speech. This was done by marking passages by the character '#' and giving every interruption a unique number. This number has to occur twice in the dialogue: once for both agents (figure 1).

4 CODING SCHEME

Goals of the current research project are to identify inherent structures and statistical relationships of dialogues and to develop a taxonomy of information dialogues based on these structures and statistical relations between move and phase transitions. Main point of interest is how information is transferred between agents. We therefore developed a coding scheme based on the coding of information and dialogue acts used to transfer information. The coding scheme is intended to be used as a tool for the analysis of the dialogues mentioned in the previous section. It is not intended for generating speech.

4.1 Outline

Dialogues are coded based on utterances. Every utterance can consist of several grammatical sentences, i.e. moves which are coded as comma separated terms. A move takes place in a certain phase of the conversation. This phase is coded as an argument of the move. Other arguments of moves are symbols used to code information, like Location, Time, etc. Information codes can have layers of subcodes with more detailed information (see figure 2). The final argument of a move is used to code interruptions.

4.2 Phases

Dialogues can be divided in several phases or subdialogues which accomplish one step in retrieving the desired information. Phases are typically a number of consecutive topic packets [10] used to complete such a step in the dialogue. Examples of subdialogues modelled by phases, are defining which information is needed or presenting the information to the client.

Phases are important aspects of our coding. Dialogue phase is a good indication of the flow of information at a certain time in the dialogue. For instance, during the Greeting phase both agents give information, while in the Query phase most information is given by the client. Another important aspect of dialogue phase is its usability as a measure of dialogue complexity. In our coding scheme, six phases can be distinguished.

Greeting (G) At the beginning of the dialogue both parties exchange greetings and often introduce themselves.

Query (Q) In this phase the client has the initiative and tries to pose his question. All parameters which contribute to the question can be altered during this phase, for instance the client can mention both the arrival time and departure place during one instance of the Query phase. In some dialogues there is no explicit question ('Tomorrow I have to be in Rotterdam'). When the given information by the client is insufficient to find a solution, the operator can elicit further information by posing supplementary questions.

Pause (P) Periods of silence occur during dialogues, indicating the operator is searching. These pauses can often be found between Query and Information phase and are frequently introduced by the operator.

Information (I) During this phase the operator provides the information asked for by the client. The operator who has the initiative, often presents the information in small pieces to make it easier for the client to memorise or write down the information. Each piece is usually acknowledged by the client.

Subquery (S) Subqueries are phases which vary only a small part of the original query. Subqueries can originate from a lack of information provided for by the client or from imprecise or wrong information provided for by the operator. Subqueries can therefore both be initiated by the operator and by the client. Operator initiated subqueries are usually supplementary questions to retrieve information from the client which has not been given and which is necessary to extract the information asked for. Client initiated subqueries are strongly related to the original query. The reason for having a Subquery phase besides the Query phase enables us to detect inadequate or incomplete information transfer. This makes it possible to develop a more precise dialogue taxonomy (see section 6).

Goodbye (B) The end of a dialogue is often introduced by an alignment, which is used to check whether both the client and operator are ready to end the conversation by exchanging thanks and saying goodbye.

Looking at these phases, in only three of them relevant information is exchanged: Query, Subquery and Information. The other phases serve social goals, because information dialogues have to observe norms and conventions of both agents' cultures.

4.3 Moves

Phases consist of one or more conversational games [3]. These games usually consist of initiations, e.g. questions or statements, followed by responses, like answers or acknowledgements. Moves are simply different kinds of initiations and responses classified according to their purposes.

Question (Que) Whenever an agent asks for a piece of information, this is coded as a question. Implicit questions (i.e. ones that start with 'I like to know ...') are also coded as a question. Information can be introduced as part of a question.

Statement (Sta) Statements are utterances with the only purpose of presenting information in response to a question posed earlier in the dialogue. Typically this information is not given in an interrogative form. Statements are often part of an answer or describe the context of a new question.

Acknowledgement (Ack) An acknowledgement is a verbal response which minimally shows that the speaker has heard the move to which it responds. It often also demonstrates the acceptance or rejection of the move which is acknowledged.

Check (Che) A check requests the partner to acknowledge information that the checker has some reason to believe, but is not entirely sure about. Typically the information to be confirmed is something which the checker believes was meant to be inferred from what the partner has said. For instance, when a client mentions Rotterdam as the destination, the operator usually checks if Rotterdam Central Station was meant as the destination of the journey.

Reconfirmation (Rec) A reconfirmation requests the partner to repeat the answer or question given. This occurs in three cases: the person who asks for reconfirmation did not hear the partner's move, the person is aware of some forgotten information, or the person does not believe a given answer.

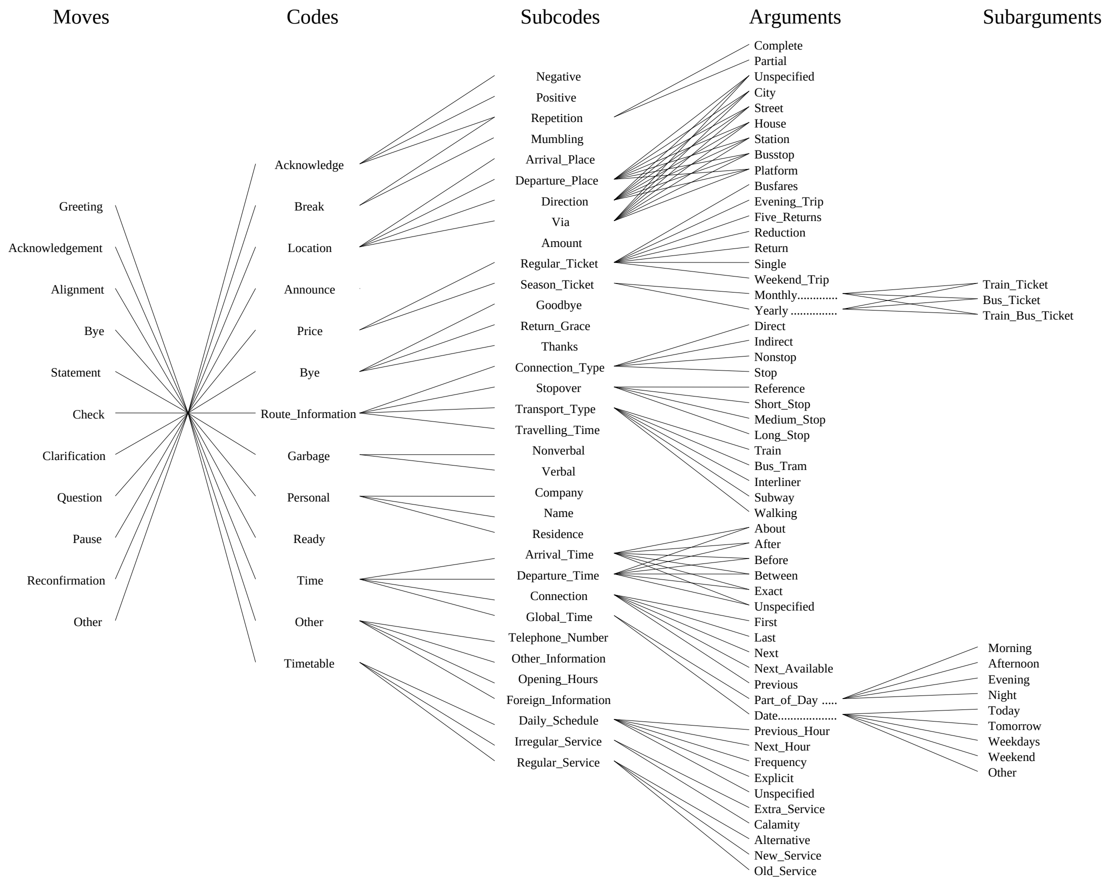

Figure 2. Outline of Coding Scheme

Clarification (Cla) A clarification is an utterance which presents a piece of additional information which was not strictly asked for by the other agent. A clarification can be a reply to a question in which case it does not present the precise information asked for. A clarification can also be given spontaneously in which case it usually is some information additional to the information presented in the previous statement.

Alignment (Ali) An alignment verifies the attention and/or agreement of the partner to move to a next phase. The most common type of alignment requests the partner to acknowledge whether the query has been answered to satisfaction or when the client wishes to drop the query and that they are both ready to move on.

Some moves are used for coding social behaviour occurring during information dialogues. These moves do not serve other goals than confirming to social behaviour normally displayed in such dialogues. In our coding scheme we use the following 'social' moves:

Greeting (Gre) Greeting is the move in which an agent greets the other agent and often introduces himself.

Bye (Bye) The bye move describes the social ritual where the client thanks the operator, who on her term expresses that she was glad to help, after which both parties say goodbye.

Pause (Pau) A pause is an indication that the operator is busy processing the query and that during the pause no information is exchanged.

Finally, we also have a code **Other (Oth)** for verbal and nonverbal garbage (like unintelligible utterances and noise).

4.4 Coding of Information

Information codes are used to code what type of information was transferred in every move described above. An example of such an information code is Location, followed by an information subcode, e.g. Departure Place. Arguments to this information code are items, like Street or City. In a few cases, arguments have subarguments. This hierarchical structure has been chosen to make automated processing of coded dialogues easier.

Arguments are sometimes suffixed by numbers indicating some piece of information. For instance, all references in a dialogue to Amsterdam could be coded as 'City(1).' Based on the coded dialogues it is possible to detect when new pieces of information are introduced or when references are made to earlier introduced information.

In this way examining information references is possible without the obstruction of the huge amount of information presented and repeated during a conversation. Translation of the coding back to the original conversation is possible, albeit without the exact words but with the correct information references.

Location (Loc) This code contains information about locations mentioned in the dialogue. This can be the arrival and departure place, but can also be intermediate stations or directions. Typical arguments for location are City, Station, etc.

Time (Tm) Time concerns the general time setting of the query. This includes arrival or departure time and date of travelling, but also relative time settings like next or last connection. Arguments for this code are typical time and date references.

Route Information (RI) Route Information covers the type of the journey to be made. This information ranges from Transport Type (e.g. Bus or Train) and Connection Type (e.g. Direct or Nonstop) to Stopovers and Travelling time.

Timetable (Tt) This code concerns all information producing insight into the general set-up of the timetable belonging to a specific journey. This can be information of a regular timetable as well as information concerning changes in regular timetables as an effect of irregularities.

Price (Pr) Price covers the costs of all types of public transport tickets. This covers regular tickets, like Single and Return tickets, as well as Reduction and Season Tickets.

Acknowledge (Ack) Acknowledge is used to code all acknowledgements in response to both questions and statements. Acknowledgements can be either Positive, Negative or Repetitions.

Other (Oth) Other represents all information which can not be coded by any of the other codes. Typical examples are telephone numbers. Social chit-chat is also coded as Other.

Three codes are used to code control information. **Announce (Ann)** marks information concerning the announcement of a Pause or Alignment. **Break (Bre)** concerns any irrelevant moves during a pause in the conversation which can be utterances to report progress or to break the silence. **Ready (Rea)** is used to get the attention of the partner, typically at the end of a pause.

Social information is represented by the following codes. **Personal (Per)** is used to model personal information, like Name and Residence. **Bye (Bye)** is used to code the final moves of a dialogue. Typical arguments are Thanks, Return Grace, and Goodbye. **Garbage (Gar)** is anything irrelevant to the goal of the dialogue.

The coding of the dialogue in figure 1 is given in figure 3.

5 RESULTS

The coding scheme was applied to a sample of 483 dialogues. We have chosen not to code 37 of them since they would not help us designing an ASP system. For instance one dialogue was about an extremely worried parent about a missing daughter. This conversation will never occur with a computer. Another example is a dialogue entirely in a foreign language. Since our speech system will work with Dutch only, this dialogue was irrelevant for our purpose. Although in theory we could have coded these dialogues, we have chosen to call these 37 dialogues 'uncodable.'

```
2: Gre(G,[],[])
1: Gre(G,[Per([Nam])],[]),
   Que(Q,[Tt([DaS([Unspecified])]),
          RI([TrT([Bus_Tram(0)])]),
          Loc([DeP([City(1)]),
               ArP([City(2)])])],[])
2: Que(S,[Tm([DeT([Unspecified])])],[])
1: Sta(Q,[Tm([DeT([About(3)])])],[])
2: Ack(Q,[Ack([Pos])],[])
2: Sta(I,[Tm([DeT([Exact(4)])]),
          RI([TrT([Bus_Tram(5)])])],[])
1: Ali(B,[Ann],[]),Bye(B,[Bye([Tha])],[])
2: Bye(B,[Bye([ReG])],[(0,1)])
1: Bye(B,[Bye([ReG,Goo])],[(0,1),(1,2)])
2: Bye(B,[Bye([Goo])],[(0,1)])
```

Figure 3. Sample coding (Information codes are abbreviated.)

It was to be expected that a larger part of the longer dialogues would be uncodable. This turned out to be true, but half the uncodable dialogues were shorter than 2 minutes and a quarter was even shorter than a minute.

The coding process has been performed by 3 men. The first 25 dialogues coded were identical for all 3 coders. After 10 and 25 dialogues there were review sessions to determine if the coding did not diverge too much. After the first 25 dialogues the remaining dialogues were distributed over the coders. During the coding of those dialogues the coders consulted each other frequently.

5.1 Phase Transitions

To get a first impression of the dialogues we started by examining the phase transitions that took place. The transitions between the phases Q, S, P, and I are eminent which was to be expected. The core of the conversations (see table 2) consists of queries, pauses used to search the databases, and by giving the information needed.

Table 2. Phase transitions

	G	Q	S	P	I	B
G	-	444	0	2	0	0
Q	0	-	181	282	267	5
S	0	199	-	55	174	0
P	1	122	69	-	262	2
I	0	75	175	99	-	443
B	0	9	9	0	38	-

A Query phase will follow the Greeting phase and just before the Bye phase information will be given in most cases as expected. The other cases are infrequent but nevertheless interesting.

The 5 cases with a QB transition are important since all theses dialogues ended after a question and were not followed by an answer. Further investigation showed that the operator could not help the client. In most cases the client did not know the exact location where he wanted to go. The public transportation information system can cope with exact addresses but not with names of buildings (like conference centres).

The cases of BQ & BS transitions are typically clients interrupting the Bye phase because they realise that they have another question. On the other hand a BI transition typically occurs when an operator interrupts the Bye phase by giving the client extra information that just popped up.

In the 2 cases where there was a GP transition the clients were not ready yet to formulate their question. They were probably distracted when they were on hold. The case of the PG transition was caused by a change of operator. The two cases of the PB transitions were quite different: in one dialogue the client interrupts the operator's search by saying that he has enough information while in the other dialogue the client took a long time to copy down the information.

5.2 Phase Automaton

For an ASP system we found the QB, BQ, BS, and the BI transitions of the infrequent cases important enough to set a threshold on 5 dialogues for a transition to be significant.

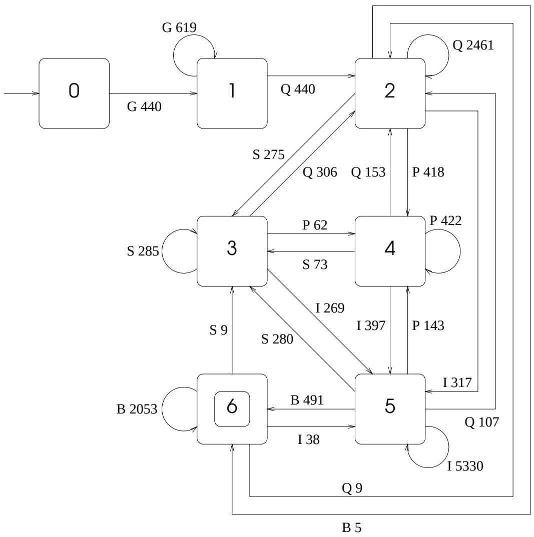

Figure 4. Phase automaton (initial state = 0, final state = 6 and transitions are labeled by phase and the number of transitions found in the sample.)

Out of the remaining cases we have constructed an automaton (see figure 4) that can accept the phases that occur in dialogues (including non-changing phases). Of all dialogues just 6 are rejected: 5 because of an illegal transition (caused by the transitions that did not reach the threshold) and in one dialogue the final state was not reached. Although tempting to believe, the states in the diagram have no direct link with the phases in a dialogue but are due to the automatic construction of the automaton.

5.3 Phase Profiles

When we look at the phase transitions of the dialogues we find that a few phase profiles (i.e. strings describing the phase transitions) occur frequently (see table 3). Most of these profiles are short:

- Short dialogues occur more frequently.
- There are only a few short profiles possible.
- Long profiles are more likely to be different.

Table 3. Phase profiles occurring more often than 10 times in the 446 coded dialogues

Phase Profiles	#	%
GQIB	51	11.4
GQPIB	35	7.8
GQPQIB	15	3.4
GQSQIB	14	3.1
GQPISIB	12	2.7
GQSQPIB	11	2.5

With respect to the last point it must be said that longer profiles can easily be clustered to similar cases.

The profiles GQIB, GQPIB, GQPQIB, and GQPISIB all have one thing in common: the client supplies all the information needed by itself such that the operator is capable to answer the question. In the profiles GQSQIB and GQSQPIB the operator asks the client to supply further information so that the operator can answer the client's question.

5.4 Moves

At this moment we have only been able to make a first investigation of all the coded dialogues. Even at the level of moves the amount of data is vast. Therefore we will only present a rough outline of the moves found in the dialogues (see [11] for a detailed description).

In the Greeting phase both the Greeting moves (5.6% of all moves) and the Acknowledgements (1.1% of all moves) are frequent. The acknowledgements are typically the ones uttered by the client as a reply to the opening greeting of the operator.

In the Query phase Statements (6.4%) are much more frequent than Questions (3.2%). This might look strange but it is not: in speech a question is often preceded with short statements describing the context of the question. The number of Acknowledgements (8.0%) is large. On the one hand the operator uses a 'ja' ('yes') to keep the conversation going, while on the other hand the client uses acknowledgements to agree with default choices made by the operator. An operator usually Checks (2.1%) when a default choice occurs. Typical examples are assuming train stations as the default destination when a client has only referred to a city.

In the Subquery phase both Questions (4.2%) and Acknowledgements (1.3%) occur. This was to be expected due to the definition of this phase. Moves in the Pause phase are Pauses (6.0%). Note that at a deeper level our Pause moves have codes for the announcing and ending the pause.

The Information phase is dominated by Statements (14.4%), most of them made by the operator and by Acknowledgements (19.0%), most of them made by the client. Typically the operator dictates and the client copies the information down and acknowledges (often repeating part of what the operator has said). Besides Statements also Checks (2.1%), Clarifications (1.6%), Reconfirmations (1.3%), and Other moves (2.3%) occur. Remarkably many of the Other moves are of a social nature; we suspect that the client is more at rest and in for a little bit of chit-chat (no more than a few utterances per dialogue).

After the final Information phase the operator often questions the client with an inquiring 'ja?' ('yes?') if he has finished his questioning. Also the client can initiate the alignment by saying that he has enough information. We have chosen to include these Alignments in the Bye phase where they appear frequently (4.1%). After the alignment the social ritual of saying Bye (9.3%) takes place. Due to technical reasons a large amount of our dialogues end with nonverbal garbage coded as Other moves (2.4%).

6 CLASSIFICATION

Although the difficulty of a dialogue is proportional to its length, it is also possible to make a second taxonomy of the difficulty of dialogues. In the first part of this section we will present this classification. Next we will present the results of applying our classification to the sample of coded dialogues. In the final part of this section we will investigate the theoretical consequences for an ASP system.

6.1 Dialogue Types

One of our objectives is to investigate the potential difficulty of an ASP system that can handle a significant portion of the dialogues. We have defined five types of dialogues. Each of these succesive types needs a more powerful ASP system to recognise the dialogues. These types do not vary with dialogue length or the complexity of sentences. Types are primarily distinguished by phase transitions and secondly certain moves like checks are used to make further distinctions. We distinguish the following dialogue types:

1. The client presents all necessary data. In the easiest form this can be regarded as filling in a question form in free order. Once all slots are filled, the operator is able to retrieve the information requested.

2. The operator continues to ask until the client has given all necessary data. Often a client has forgotten to fill in all slots and the operator keeps on asking for the missing slots. In many cases the client is not aware what information is necessary for the operator to be able to retrieve an answer from the information system.

3. The operator helps with suggestions. Sometimes the client does not know the exact location where he wants to go to. Often the operator can help out by suggesting a place nearby.

4. There is insufficient data for the operator. The operator cannot help the client with his question and the dialogue stops without a travel plan.

5. Miscellaneous. Actually, this is none of the above. It can also be seen as the type of conversation that would not be processed by a computer (at least at this stage).

With our coding scheme we could recognise these types as follows (disregarding the Pause phase):

1. These dialogues go directly from the Query phase to the Information phase and they do not belong to type 3. These dialogues do not have a Subquery phase in the Query phase.

2. Just as type 1, but with a Subquery phase in the Query phase.

3. These dialogues contain a Check move initiated by the operator in the Query phase and in the Check new information is presented.

4. These dialogues do not have an Information phase before the Bye phase but a Query phase.

5. These were the uncoded dialogues.

Using several tools we were able to establish the type of all coded dialogues (see table 4). It was a bit surprising to see that the number of conversations with help of the operator was so small and that the number of conversations where the client's question remains unanswered was even smaller.

Table 4. Dialogue types

Type	#	%
1	134	28
2	291	60
3	17	4
4	5	1
5	37	8

An ASP system that works with question forms (like the one described in [9]) could in theory handle all cases of type 1 and 2. This leads to an upper bound of 88% of all conversations that could be processed by form based systems. In practice this upper bound is far too optimistic.

6.2 Parsing

Spoken text does not resemble written text. Most spoken sentences can be regarded as a concatenation of short sentence fragments of a relatively easy structure. The speaker hopes that the listener is able to understand the meaning of the spoken sentence. In most cases the entire sentence is not grammatical but the fragments are fairly grammatical. Ambiguity is less a problem in spoken text than it is in written text. To give two examples: subordinate clauses and recursive preposition phrases are far less frequently used in spoken text than in written text.

A language processing system has two major subsystems:

- A subsystem responsible for the lexical analysis: a tokeniser will transforms a sentence into separate words from which the meaning is known.

- A subsystem responsible for the grammatical analysis: a parser will retrieve the semantics of a sentence. Normally the parser will make use of the tokeniser.

Most parsers can only handle grammatical sentences. However, these parsers are not well suited for spoken language:

- The sentence fragments and their concatenations are often non-grammatical.

- A tokeniser for spoken language is a speech recogniser. With the current state of art only a limited number of words can be recognised and so there will be many words for which the speech recogniser must pass.

- A speech recogniser will often misunderstand words that sound similar.

This leads quite naturally to a parser that is capable of coping with non-grammatical sentences, missing words, and incorrect words: a robust parser (see [6, 8]). Many robust parsers first try to parse the sentence in a non-robust way and when this fails they incrementally try to adjust the partial parse tree or sentence in such a way that parsing can continue. Performance deteriorates with the number of adjustments. A sentence with more than a couple of mistakes will hardly be parsable in real time.

The bottleneck in this scheme will be the speech recogniser [7]. Even with a near perfect speech recogniser we cannot process more than $\frac{2}{3}$ of the dialogues. Dialogues about train information only are favourable. The vocabulary for such dialogues is much smaller: the number of stations is about two orders smaller than the total amount of proper names (a.o. streets and cities) accessible in the current information system. Also quite important is that on average these dialogues are much shorter.

7 FUTURE RESEARCH

Since we will work on an ASP demonstrator in the long term, we have planned several subgoals along the way. As a first step we have started two independent studies.

The first one, as described in this paper, was to analyse the dialogues between clients and operators using the current system. Although we already have a substantial amount of statistics, we still need more statistics on a more detailed level. To get usable statistics we need to code more dialogues. At the current time we have manually coded the dialogues. We plan to work on a way to do this automatically or semi-automatically (i.e. human assisted).

The second study is an experiment [11] where 409 people were instructed to travel between two places at a certain time. The persons were asked to use the current system and to use an ASP system. The persons were not aware that the ASP system was actually a simulated system operated by a human. The travel instructions given to the participants were made in such a way that they had to formulate their own questions to the ASP system. The simulated system was not the same for all people, we variated the way the system worked so that we were able to establish which type of human-machine dialogues were best appreciated.

As a second step we plan to evaluate available ASP systems by using our coding scheme and classification. Together with the results obtained from the research described in this paper and the second study, we plan to design a suitable dialogue structure for a dialogue manager that can be used in the ASP demonstrator. Concurrently to this we will work on a speech recogniser.

In the final step we will try to fit all into a prototype that will show the strengths of the techniques used. If this prototype works satisfactory, it will be reimplemented more efficiently and in parallel by others such that it can be used by OVR. It is hoped that the ASP system will supplement the current operators such that their workload can be reduced. We hope that the ASP system will take the easier client questions, while the operators can concentrate on the more interesting questions.

References

[1] J. Alexandersson and N. Reithinger. Designing the dialogue component in a speech translation system, A corpus based approach. In *Corpus-based Approaches to Dialogue Modelling*, 9th Twente Workshop on Language Technology, pages 35–43. University of Twente, 1995.

[2] H.C. Bunt. Dynamic interpretation and dialogue theory. In *The Structure of Multimodal Dialogue*, volume 2. John Benjamins Publishing Company, Amsterdam, 1995.

[3] J. Carletta, A. Isard, S. Isard, J. Kowtko, G. Doherty-Sneddon, and A. Anderson. The coding of dialogue structure in a corpus. In *Corpus-based Approaches to Dialogue Modelling*, 9th Twente Workshop on Language Technology, pages 25–34. University of Twente, 1995.

[4] P.R. Cohen. Models of dialogue. In *Cognitive Processing for Vision & Voice*, 4th NEC Research Symposium, pages 181–204. NEC & SIAM, 1994.

[5] N. Dahlbäck. Kinds of agents and types of dialogues. In *Corpus-based Approaches to Dialogue Modelling*, 9th Twente Workshop on Language Technology, pages 1–11. University of Twente, 1995.

[6] D. van der Ende. Robust parsing: An overview. Technical Report Memoranda Informatica 95-03, University of Twente, Department of Computer Science, 1995.

[7] E. Keller, editor. *Fundamentals of Speech Synthesis and Speech Recognition, Basic Concepts, State-of-the-Art and Future Challanges*. John Wiley & Sons, 1994.

[8] R. Kiezebrink and J.P.M. de Vreught. An annotated bibliography on robust parsing and related topics. Technical Report CS 96-58, Delft University of Technology, 1996.

[9] M. Oerder and H. Aust. A realtime prototype of an automatic inquiry system. In *International Conference on Spoken Language Processing*, pages 703–706, 1994.

[10] M.M.M. Rats. *Topic Management in Information Dialogues*. Ph.D. thesis, Tilburg University, 1996.

[11] L.J.M. Rothkrantz, R.J. van Vark, J.P.M. de Vreught, J.W.A. van Wees, and H. Koppelaar. Automatische spraakherkenning en OVR-dialogen. Technical report, Delft University of Technology, Knowledge Based Systems, 1996. in Dutch.