User studies and the design of Natural Language Systems

Steve Whittaker and Phil Stenton
Hewlett-Packard Laboratories
Filton Road, Bristol BS12 6QZ, UK.
email: sjw@hplb.hpl.hp.com

Abstract

This paper presents a critical discussion of the various approaches that have been used in the evaluation of Natural Language systems. We conclude that previous approaches have neglected to evaluate systems in the context of their use, e.g. solving a task requiring data retrieval. This raises questions about the validity of such approaches. In the second half of the paper, we report a laboratory study using the Wizard of Oz technique to identify NL requirements for carrying out this task. We evaluate the demands that task dialogues collected using this technique, place upon a prototype Natural Language system. We identify three important requirements which arose from the task that we gave our subjects: operators specific to the task of database access, complex contextual reference and reference to the structure of the information source. We discuss how these might be satisfied by future Natural Language systems.

1 Introduction

1.1 Approaches to the evaluation of NL systems

It is clear that a number of different criteria might be employed in the evaluation of Natural Language (NL) systems. It is also clear that there is no consensus on how evaluation should be carried out [RQR*88, GM84]. Among the different criteria that have been suggested are (a) Coverage; (b) Learnability; (c) General software requirements; (d) Comparison with other interface media. **Coverage** is concerned with the set of inputs which the system should be capable of handling and one issue we will discuss is how this set should be identified. **Learnability** is premised on the fact that complete coverage is not forseeable in the near future. As a consequence, any NL system will have limitations and one problem for users will be to learn to communicate within such limitations. Learnability is measured by the ease with which new users are able to identify these coverage limitations, and exploit what coverage is available to carry out their task. The **general software criteria** of importance are speed, size, modifiability and installation and maintenance costs. **Comparison** studies have mainly required users to perform the same task using either a formal query language such as SQL or a restricted natural language and evaluated one against the other on such parameters as time to solution or number of queries per task[SW83, JTS*85]. Our discussion will mainly address the problem of coverage: we shall not discuss these other issues further.

Our concern here will be with interactive NL interfaces and not other applications of NL technology such as MT or messaging systems. Interactive interfaces are not designed to be used in isolation, rather, they are intended to be connected to some sort of backend system, to improve access to that system. Our view is that NL systems should be evaluated with this in mind: the aim will be **to identify the NL inputs which a typical user would want to enter in order to utilise that backend system to carry out a representative task.** By representative task we mean the class of task that the back-end system was designed to carry out. In the case of databases, this would be accessing or updating information. For expert systems it might involve identifying or diagnosing faults.

1.1.1 Test suites

One method that is often used in computer science for the evaluation of systems is the use of test suites. For NL systems the idea is to generate a corpus of sentences which contains the major set of syntactic, se-

mantic and pragmatic phenomena the system should cover [BB84, FNSW87]. One problem with this approach is how we determine whether the test set is complete. Do we have a clear notion of what constitute the major phenomena of language so that we can generate test sentences which identify whether these have been analysed correctly? Theories of syntax are well developed and may provide us with a good taxonomy of syntactic phenomena, but we do not have similar classifications of key pragmatic requirements. There are two reasons why current approaches may fail to identify the key phenomena. Current test sets are organised on a single-utterance basis, with certain exceptions such as intersentential anaphora and ellipsis. Now it may be that more complex discourse phenomena such as reference to dialogue structure arise when systems are being used to carry out tasks, because of the need to construct and manipulate sets of information [McK84]. In addition, context may contribute to inputs being fragmentary or telegraphic in style. Unless we investigate systems being used to carry out tasks, such phenomena will continue to be omitted from our test suites and NL systems will have to be substantially modified when they are connected to their backend systems. Thus we are not arguing against the use of test suites in principle but rather are attempting to determine what methodology should be used to design such test suites.

1.1.2 Field studies

In field studies, subjects are given the NL interface connected to some application and encouraged to make use of it. It would seem that these studies would offer vital information about target requirements. Despite arguments that such studies are highly necessary [Ten79], few systematic studies have been conducted [Dam81, JTS*85, Kra80]. The problem here may be with finding committed users who are prepared to make serious use of a fragile system.

A major problem with such studies concerns the robustness of the systems which were tested and this leads to difficulties in the interpretation of the results. This is because a fragile system necessarily imposes limitations on the ways that a user can interact with it. We cannot therefore infer that the set of sentences that users input when they have adjusted to a fragile system, reflects the set of inputs that they would wish to enter given a system with fewer limitations. In other words we cannot infer that such inputs represent the way that users would ideally wish to interact using NL. The users may well have been employing strategies to communicate within the limitations of the system and they may therefore have been using a highly restricted form of English. Indeed the existence of strategies such as paraphrasing and syntax simplification when a query failed, and repetition of input syntax when a query succeeded has been documented [Tho80, WW89].

Since we cannot currently envisage a system without limitations, we may want to exploit this ability to learn system limitations, nevertheless the existence of such user strategies does not give us a clear view of what language might have been used in the absence of these limitations.

1.1.3 Pen and paper tasks

One technique which overcomes some of the problems of robustness has been to use pen and paper tasks. Here we do not use a system at all but rather give subjects what is essentially a translation task [JTS*85, Mil81]. This technique has also been employed to evaluate formal query languages such as SQL. The subjects of the study are given a sample task: *A list of alumni in the state of California has been requested. The request applies to those alumni whose last name starts with an S. Obtain such a list containing last names and first names.* When the subjects have generated their natural language query, it is evaluated by judges to determine whether it would have successfully elicited the information from the system.

This approach avoids the problem of using fragile systems, but it is susceptible to the same objections as were levelled at test suites: a potential drawback with the approach concerns the representativeness of the set of tasks the users are required to do when they carry out the translation tasks. For the tasks described by Reisner, for example, the queries are all one shot, i.e. they are attempts to complete a task in a single query [Rei77]. As a result the translation problems may fail to test the system's coverage of discourse phenomena.

1.1.4 Wizard of Oz

A similar technique to pen and paper tasks has been the use of a method called the "Wizard of Oz" (henceforth WOZ) which also avoids the problem of the fragility of current systems by simulating the operation of the system rather than using the system itself.

In these studies, subjects are told that they are interacting with the computer when in reality they are linked to the Wizard, a person simulating the operation of the system, over a computer network.

In Guindon's study using the WOZ technique, subjects were told they were using an NL front-end to a knowledge-based statistics advisory package [GSBC86]. The main result is a counterintuitive one. These studies suggest that people produce "simple language" when they believe that they are using an NL interface. Guindon has compared the WOZ dialogues of users interacting with the statistics package, to informal speech, and likened them to the simplified register of "baby talk" [SF77]. In comparison with informal speech, the dialogues have few passives, few pronouns and few examples of fragmentary speech.

One problem with the research is that it has been descriptive: It has chiefly been concerned with demonstrating the fact that the language observed is "simple" relative to norms gathered for informal and written speech and the results are expressed at too general a level to be useful for system design. It is not enough to know, for example, that there are fewer fragments observed in WOZ type dialogues than in informal speech: it is necessary to know the precise characteristics of such fragments if we are to design a system to analyse these when they occur. Despite this, our view is that WOZ represents the most promising technique for identifying the target requirements of an NL interface. However, to avoid the problem of precision described above, we modified the technique in one significant respect. Having used the WOZ technique to generate a set of sentences that users ideally require to carry out a database retrieval task, we then input these sentences into a NL system linked to the database. The target requirements are therefore evaluated against a version of a real system and we can observe the ways in which the system satisfies, or fails to satisfy, user requirements.

1.2 The current study

The current study therefore has two components: the first is a WOZ study of dialogues involved in database retrieval tasks. We then take the recorded dialogues and map them onto the capabilities of an existing system, HPNL [NP88] to look at where the language that the users produce goes beyond the capabilities of this system. The results we present concern the first phase of such an analysis in which we discuss the set of words that the system failed to analyse. We discuss semantics and pragmatics only insofar as they are reflected in individual lexical items. This is of some importance, given the lexical basis of the HPNL system. It must also be noted that the evaluation took place against a prototype version of HPNL. Many of the lexical errors we encountered could be removed with a trivial amount of effort. Our interest was not therefore in the absolute number of such errors, but rather with the general classes of lexical errors which arose. We present a classification of such errors below.

The task we investigated was database retrieval. This was predominantly because this has been a typical application for NL interfaces. Our initial interest was in the target requirements for an NL system, i.e. what set of sentences users would enter if they were given no constraints on the types of sentences that they could input. The Wizard was therefore instructed to answer all questions (subject to the limitation given below). We ensured that this person had sufficient information to answer questions about the database, and so in principle, the system was capable of handling all inputs.

The subjects were asked to access information from the "database" about a set of paintings which possessed certain characteristics. The database contained information about Van Gogh's paintings including their age, theme, medium, and location. The subjects had to find a set of paintings which together satisfied a series of requirements, and they did this by typing English sentences into the machine. They were not told exactly what information the database contained, nor about the set of inputs the Natural Language interface might be capable of processing.

2 Method

2.1 Subjects

The 12 subjects were all familiar with using computers insofar as they had used word processors and electronic mail. A further 5 of them had used office applications such as spreadsheets or graphics packages. Of the remainder, 4 had some experience with using databases and one of these had participated in the design a database. None of them was familiar with the current state of NL technology.

2.2 Procedure

The experimenter told the subjects that he was interested in evaluating the efficiency of English as a medium for communicating with computers. He told them that an English interface to a database was running on the machine and that the database contained information about paintings by Van Gogh and other artists. In fact this was not true: the information that the subjects typed into the terminal was transmitted to a person (The Wizard) at another terminal who answered the subject's requests by consulting paper copies of the database tables.

The experimenter then gave the details of the two tasks. Subjects were told that they had to find a set of paintings which satisfied several requirements, where a requirement might be for example that (a) all the paintings must come from different cities; or (b) they must all have different themes. Having found this set, they had then to access particular information about the set of pictures that they had chosen, e.g the paint medium for each of the pictures chosen.

Our interest was in the target set of queries input by people who wanted to use the system for database access. We therefore gave the Wizard instructions to answer all queries regardless of linguistic complexity. There was however one exception to this rule: each task was expressed as a series of requirements and one possible strategy for the task was to enter all these requirements as one long query. If the Wizard had answered this query then the dialogue would have been extremely short, i.e it would have been one query and a response which was the answer to the whole task. To prevent this, the Wizard was told to reply to such long queries by saying *Too much information to process.* There were no other constraints on the type of input that the Wizard could process and answers were given to all other types of query.

Subject and Wizard both used HP-Unix Workstations and communicated by writing in networked X windows. The inputs of both subject and Wizard were displayed in a single window on each of the machines with the subject's entries presented in lower case and the Wizard's in upper case, so the contents of the display windows on both machines were identical. To avoid teaching the subjects skills like scrolling, we also provided them with hard copy output of the whole of the interaction by printing the contents of the windows to a printer next to the subject's machine. If they wanted to refer back to much earlier in the dialogue, the subjects could consult the hard copy.

3 Results

3.1 Preliminary analysis and filtering

This analysis is concerned with user input and so the Wizard's responses are not considered here. We began by taking all the 384 subject utterances, entering them into the NL prototype and observing what analysis the system produced. We found that by far the largest category of errors was unknown words, so we began by analysing the total of 401 instances of 104 unknown words.

Our interest here lay in the influence of the task on language use so we focus on 3 classes of unknown words which demonstrate this in different ways: these were operators and explicit reference to set properties; references to context; and references to the information source.

3.1.1 Operators and the explicit specification of set properties

The task of database access involves the construction and manipulation of answer sets with various properties.

The unknown words that were used for set construction and manipulation were mainly verbs. These we called **operators**. They can be further subclassified into verbs which were used to **select** sets, those which were used to **permute** already constructed sets and those which **operate over a set of queries**. The majority of operators invoked simple set selection: these included for example, *state* and *tell.* There were also instances of indirect requests for selection, e.g. *need* and *want.* Subjects tried to permute the presentation of sets by using words like *arrange.* Finally queries such as *All the conditions from now on will* **apply** *to ...* show there were verbs which operated over sets of queries.

A second way in which these set manipulation operations appeared was in the subjects' explicit reference to the fact that they were constructing sets with specific properties. *Find paintings that satisfy the following* **criteria** ... was an example of this.

Altogether operators and explicit reference to set

properties occurred on 102 occasions which accounted for 25% of the unknown words.

3.1.2 References to context

The task could not be accomplished in one query so we expected that this would necessitate our subjects making reference to previous queries. We therefore went on to analyse those unknown words that required information from outside the current query for their interpretation. Among the unknown words which relied upon context, we distinguished between what we called **pointers** (N = 42 instances) and **exclusion operators** (N = 21 instances). Together they accounted for 16% of unknown words.

Pointers signalled to the listener that the reference set lay outside the current utterance. These could be further subdivided according to whether or not they pointed **forwards**, e.g. *Give me the dates of the* **following** *paintings* ... or **backwards** in the dialogue, e.g. *previous* and *above*. There were two instances of forwards pointers *following* and *now on*.

The backwards pointers could be subclassified according to how many previous answer sets they referred to. The majority referred to a single answer set and this was most often the one generated by the immediately prior query. Other pointers referred to a number of prior answer sets, which could scope as far back as the beginning of the current subdialogue, or even the beginning of the whole dialogue.

Exclusion operators applied to sets created earlier in the dialogue. They served to exclude elements of these sets from the current query. The simplest examples of this occurred when people had (a) identified a set previously; (b) they had then selected a subset of this original set; and (c) they wanted all or part of the set of the original set which had not been selected by the second operation. These included words like *another* and *more*, as in *Give me 10* **more** *Van Gogh paintings.*

A more complex instance of this type of exclusion was when the word was used, not to exclude subsets from sets already identified, but to exclude the attributes of the items in the excluded subsets, e.g. *Find me a painting with a theme that is* **different** *from those already mentioned.* Here the system has first to generate the set of paintings already mentioned, then it has to generate their themes and then finally it has to find a painting whose theme is different from the set of themes already identified.

3.1.3 References to the information source

Our subjects believed that they were interacting with a real information source, in this case a database, also seemed to affect their language use. We found 19 (5% of all unknown words) which seemed to refer to the database and its structure directly.

There were words which seemed to refer to **field names** in the database, e.g. *categories* and *information*, e.g. *What* **information** *on each painting is there?* There were also words which seemed to refer to **values within a field**, e.g. *types* as in *List the media* **types**. In addition, there were references to the **ordering** of entities, e.g. *first* or *second*, as in *What is the* **first** *painting in your list?*. Finally, there were words which referred to the **general scope or properties** of the database: e.g. *database* and *represented*, e.g. *What different paint media are* **represented**?.

There were also 3 occasions on which reference is made both to database structure and to context. These are the instances of *next* being used to access entities in a column but also referring to context. The utterance *List* **next** *10 paintings*, references 10 items in the sequence that they appear in the database, but excludes the 10 items already chosen. Finally there was one instance of a question which would have required inferencing based on the structure of the information source, *Is a portrait the* **same** *as a self portrait?*. Here the question was about the type relation.

4 Conclusions

This paper had two objectives: the first was to evaluate the use of the WOZ technique for assessing NL systems and the second was to investigate the effect of task on language use.

One criticism we made of both test suites and tasks using pen and paper, was that they may attempt to evaluate systems against inadequate criteria. Specifically they may not evaluate the adequacy of NL systems when users are carrying out tasks with specific software systems. The unknown words analysis seems to bear this out: we found 3 classes of unknown words which occurred only because our users were doing a task. Firstly our users wanted to carry out operations

involving the selection and permutation of answer sets and make explicit reference to their properties. Secondly, we found that our subjects wanted to use complex reference to refer back to previous queries in order to refine those queries, or to exclude answers to previous queries from their current query. Finally, we found that users attempted to use the structure of the information source, in this case the database, in order to access information. Together these 3 classes accounted for 45% of all unknown words. We believe that whatever the task and software, there will always be instances of operators, context use and reference to the information source. It would therefore seem that coverage of these 3 sets of phenomena is an important requirement for any NL interface to an application. The fact that other evaluation techniques may not have detected this requirement is, we believe, a vindication of our approach. An exception to this is the work of Cohen et al. [CPA82] who point to the need for retaining and tracking context in this type of application.

Of course there are still problems with the WOZ technique. One such problem concerns the task representativeness and a difficulty in designing this study lay in the selection of a task which we felt to be typical of database access. Clearly more information from field studies would be useful in helping to identify prototypical database access tasks.

A second problem lies in the interpretation of the results with respect to the classification and frequency of the unknown word errors: how frequently must an error occur if it is to warrant system modification? For example, references to the information source accounted for only 5% of the errors and yet we believe this is an interesting class of error because exploiting the structure of the database was a useful retrieval tactic for some users. The frequency problem is not specific to this study, but is an instance of a general problem in computational linguistics concerning the coverage and the range of phenomena to which we address our research. In the past, the field has focussed on the explanation of theoretically interesting phenomena without much attention to their frequency in naturally occurring speech or text. It is clear, however, that if we are to be successful in designing working systems, then we cannot afford to ignore frequently occurring but theoretically uninteresting phenomena such as punctuation or dates. This is because such phenomena will probably have to be treated in whatever application we design. Frequency data may also be of real use in determining priorities for system improvement.

As a result of using our technique, we have identified a number of unknown words. How should these words be treated? Some of the unknown words are synonyms of words already in the system. Here the obvious strategy is to modify the NL system by adding these. In other cases, system modification may not be possible because linguistic theory does not have a treatment of these words.h In these circumstances, there are three possible strategies for finessing the problem. The first two involve encouraging users to avoid these words, either by generating co-operative error messages to enable the user to rephrase the query and so avoid the use of the problematic word [Adg88, Ste88] or by user training. The third strategy for finessing the analysis of such words is to supplement the NL interface with another medium such as graphics, and we will describe an example of this below.

We believe that the use of such finessing strategies will be important if NL systems are to be usable in the short term. Our data suggests that certain words are used frequently by subjects in doing this task. It is also clear that computational linguistics has no treatment of these words. If we wish to build a system which will enable our users to carry out the task, we must be able to respond in some way to such inputs. The above techniques may provide the means to do this, although the use of such strategies is still an under-researched area.

For the unknown words encountered in this study, of the **operators**, many can be dealt with by simple system modification because they are synonyms of *list* or *show*. Within the class of operators, however, it would seem that new semantic interpretation procedures would have to be defined for verbs like *arrange* or *order*. These would involve two operations, the first would be the generation of a set, and the second the sorting of that set in terms of some attribute such as age or date. The unknown words relating to explicit reference to set properties would not be difficult to add to the system, given that they can be paraphrased as relative clauses. For example, the sentence *Find Van Gogh paintings to* **include** *four different themes* can be paraphrased as *Find Van Gogh paintings that have different themes.*

The **context** words present a much more serious problem. Current linguistic theory does not have treatments of words like *previously* or *already*, in terms of how these scope in dialogues. On some occasions, these are used to refer to the immediately prior query only, whereas on other occasions they

might scope back to the beginning of the dialogue. In addition, words like *more* or *another* present new problems for discourse theory in that they require extensional representations of answers: Given the query *Give me 10 paintings* followed by *Now give me 5 more paintings*, the system has to retain an extensional representation of the answer set generated to the first query, if it is to respond appropriately to the second one. Otherwise it will not have a record of precisely which 10 paintings were originally selected, so that these can be excluded from the second set. This extensional record would have to be incorporated into the discourse model.

One solution to the dual problems presented by context words is again to either finesse the use of such words or to use a mixed media interface of NL and graphics. If users had the answers to previous queries presented on screen, then the problems of determining the reference set for phrases like *the paintings already mentioned* could be solved by allowing the users to click on previous answer sets using a mouse, thus avoiding the need for reference resolution.

For the **references to the information source**, it would not be difficult to modify the system so it could analyse the majority of the the specific instances recorded here, but it is not clear that all of them could have been solved in this way, especially those that require some form of inferencing based on the database structure.

There are also a number of unknown words in the data that have not been discussed here, because these did not directly arise from the fact that our users were carrying out a task. Nevertheless, the set of strategies given above is also relevant to these. Just as with the task specific words, there are a number of words which can be added to the system with relatively little effort. The system can be modified to cope with the majority of the open class unknown words, e.g. common nouns, adjectives, and verbs, many of which are simple omissions from the domain-specific lexicon. Some of the closed class words such as prepositions and personal pronouns may also prove straightforward to add.

There are also a number of these words which did not arise from the task, which are more difficult to add to the system. This is true for a few the open class words domain-independent words, including adjectives like *same* and *different*. The majority of the closed class words, may also be difficult to add to the system, including superlatives and various logical connectives, *then, neither*, some quantifiers, e.g. *only*, as well as words which relate to the control of dialogue such as *right* and *o.k.*. These words indicate genuine gaps in the coverage of the system. For these difficult words, it might necessary to finesse the problem of direct analysis.

In conclusion, the WOZ technique proved successful for NL evaluation. We identified 3 classes of task based language use which have been neglected by other evaluation methodologies. We believe that these classes exist across applications and tasks: For any combination of application and task, specific operators will emerge, and support will have to be provided to enable reference to context and information structure. In addition, we were able to suggest a number of strategies for dealing with unknown words. For certain words, NL system modification can be easily achieved. For others, different strategies have to be employed which avoid direct analysis of these words These finessing strategies are important if NL systems are to usable in the short term.

5 Acknowledgements

Thanks to Lyn Walker, Derek Proudian, and David Adger for critical comments.

References

[Adg88] David Adger. *Heuristic input redaction* Technical Report, Hewlett-Packard Laboratories, Bristol, 1988.

[BB84] Madeleine Bates and Robert J. Bobrow What's here, what's coming, and who needs it. In Walter Reitman, editor, *Artificial Intelligence Applications for Business*, pages 179–194, Ablex Publishing Corp, Norwood, N.J., 1984.

[CPA82] Phillip R. Cohen, C. Raymond Perrault and James F. Allen. Beyond question answering. In Wendy Lehnert and Martin Ringle, editors, *Strategies for Natural Language Processing*, pages 245–274 Lawrence Erlbaum Ass. Inc, Hillsdale N.J., 1982.

[Dam81] Fred J. Damerau. Operating statistics for the transformational question answering system. *American Journal of Computational Linguistics*, 7:30–42, 1981.

[FNSW87] Daniel Flickinger, John Nerbonne, Ivan Sag, and Thomas Wasow. Towards evaluation of nlp systems. 1987. Presented to the 25th Annual Meeting of the Association for Computational Linguistics.

[GM84] G. Guida and G. Mauri. A formal basis for performance evaluation of natural language understanding systems. *American Journal of Computational Linguistics*, 10:15–29, 1984.

[GSBC86] Raymonde Guindon, P. Sladky, H. Brunner, and J. Conner. The structure of user-adviser dialogues: is there method in their madness? In *Proc. 24st Annual Meeting of the ACL, Association of Computational Linguistics*, pages 224–230, 1986.

[JTS*85] Matthias Jarke, Jon A. Turner, Edward A. Stohr, Yannis Vassiliou, Norman H. White, and Ken Michielsen. A field evaluation of natural language for data retrieval. *IEEE Transactions on Software Engineering*, SE-11, No.1:97–113, 1985.

[Kra80] Jurgen Krause. Natural language access to information systems: an evaluation study of its acceptance by end users. *Information Systems*, 5:297–318, 1980.

[McK84] Kathleen R. McKeown. Natural language for expert systems: comparisons with database systems. In *COLING84: Proc. 10th International Conference on Computational Linguistics. Stanford University, Stanford, Ca.*, pages 190–193, 1984.

[Mil81] L. A. Miller. Natural language programming: styles, strategies and contrasts. *IBM Systems Journal*, 20:184 – 215, 1981.

[NP88] John Nerbonne and Derek Proudian. *The HPNL System Report.* Technical Report STL-88-11, Hewlett-Packard Laboratories, Palo Alto, 1988.

[Rei77] Phyllis Reisner. Use of psychological experimentation as an aid to development of a query language. *IEEE Transactions on Software Engineering*, SE-3, No.3:219–229, 1977.

[RQR*88] W. Read, A. Quilici, J. Reeves, M. Dyer, and E. Baker. Evaluating natural language systems: a sourcebook approach. In *COLING88: Proc. 12th International Conference on Computational Linguistics*, Budapest, Hungary, 1988.

[SF77] C. Snow and C. A. Ferguson. *Talking to children.* Cambridge University Press, 1977.

[Ste88] Phil Stenton. *Natural Language: a snapshot taken from ISC January 1988.* Technical Report HPL-BRC-TR-88-022, Hewlett-Packard Laboratories, Bristol, U.K., 1988.

[SW83] Duane W. Small and Linda J. Weldon. An experimental comparison of natural and structured query languages. *Human Factors*, 25(3):253–263, 1983.

[Ten79] Harry R. Tennant. *Evaluation of Natural Language Processors.* PhD thesis, University of Illinois Urbana, 1979.

[Tho80] Bozena Henisz Thompson. Linguistic analysis of natural language communication with computers. In *COLING80: Proc. 8th International Conference on Computational Linguistics. Tokyo*, pages 190–201, 1980.

[WW89] Marilyn Walker and Steve Whittaker. *When Natural Language is Better than Menus: A Field Study.* Technical Report, Hewlett Packard Laboratories, Bristol, England, 1989.