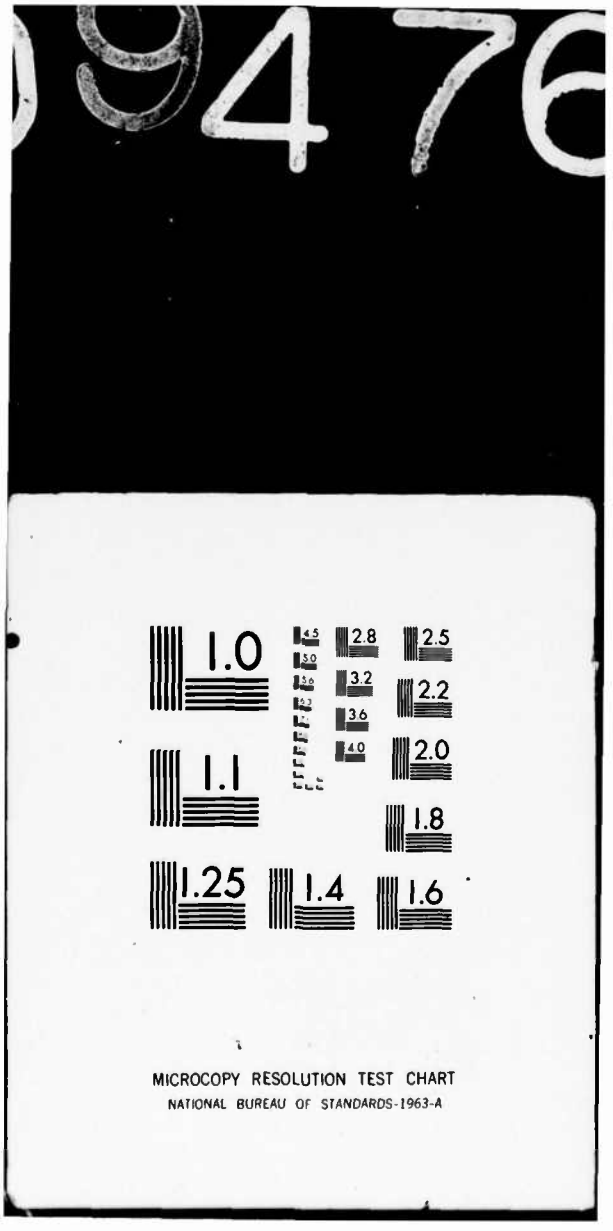
AD-A094 762 ILLINOIS UNIV AT URBANA-CHAMPAIGN COORDINATED SCIENCE LAB F/G 5/2
EVALUATION OF NATURAL LANGUAGE PROCESSORS.(U)
NOV 80 H R TENNANT N00014-75-C-0612
UNCLASSIFIED T-103 NL


9476
1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

REPORT T-103 NOVEMBER, 1980

COORDINATED SCIENCE LABORATORY

LEVEL II

13



# EVALUATION OF NATURAL LANGUAGE PROCESSORS

HARRY RALPH TENNANT







UNIVERSITY OF ILLINOIS - URBANA, ILLINOIS

(13)

(6) EVALUATION OF NATURAL LANGUAGE PROCESSORS

By



Ralph

(10) Harry Tennant

(11) Mar 80

(12) 1255

(14) T-103

(9) Doctoral thesis,



(15) This work was supported ~~in part~~ by the Office of Naval Research under contract N00014-75-C-0612 and in part by Texas Instruments, Incorporated.

# EVALUATION OF NATURAL LANGUAGE PROCESSORS

Harry Ralph Tennant, Ph.D.
Department of Computer Science
University of Illinois at Urbana-Champaign

Despite a large amount of research on developing natural language understanding programs, little work has been done on evaluating their performance or potential. The evaluations that have been done have been unsystematic and incomplete. This has lead to uncertainty and confusion over the accomplishments of natural language processing research.

The lack of evaluation can be primarily attributed to the difficulty of the problem. The desired behavior of natural language processors has not been clearly specified. Partial progress toward the eventual goals for natural language processors has not been delineated, much less measured.

This thesis attempts to clarify some of the difficulties behind evaluating the performance of natural language processors. It also proposes an evaluation method that is designed to be systematic and thorough. The method relies on considering a natural language processor from three viewpoints in the light of several taxonomies of issues relevant to natural language processing. Finally, an evaluation is described of PLANES, a natural language database query system.

# EVALUATION OF NATURAL LANGUAGE PROCESSORS

BY

HARRY RALPH TENNANT

B.S., University of Illinois at Chicago Circle, 1973
M.S., University of Illinois at Chicago Circle, 1977

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1981

Urbana, Illinois

## Acknowledgements

This thesis, like all theses, is the result of many small steps in coming to understand a problem. Few of these steps were taken alone. The great majority are taken with others. The help they offered came in many forms, such as encouragement, shared insight, inspiration or a sympathetic ear over beers. I would like to thank some of those stepped with me.

Dave Waltz provided a congenial atomosphere for research and many useful pieces of advice. I particularly admire his courage in allowing PLANES to be the first natural language processor to be subjected to thorough scrutiny.

Gary Brooks' unflagging enthusiasm and friendly nature were always a mystery. How could he work so hard but still be so congenial? Somewhat burned out at times perhaps, but amiable nevertheless.

Tim Finin provided many afternoons of stimulating discussion and several years of friendship. I recall, for example, our mutual sense of satisfaction when we first found something that KL-ONE couldn't represent.

Brad Goodman is everything a good friend ought to be. Brad was always the first to volunteer to critique a draft. If something needed doing, Brad would willingly step forward.

Afternoons at the bars with George Hadden not only refreshed the body, but revitalized the spirit. George's patience and well balanced advice made what seemed to be impenetrable obsticles appear as the minor obstructions they really were.

I have a great deal of respect for Gene Lewis. Although I successfully resisted his attempts to make a linguist out of me, I appreciate his efforts and motives. Gene also is the most entertaining storyteller I have ever met.

Bill Mann and others at ISI provided some insightful criticism on the course of my work. Its course was greatly influenced by their advice.

Others at the Coordinated Science Lab were helpful and supportive during my years there. Lois Boggess, Doug Dankel, Jeff Gibbons, Paul Rutter and Tse-Wah Wong.

Julie Tennant has supported me (in all ways!) throughout my interminable graduate career. One could not hope for more encouragement from a spouse. She even understood when I

didn't finish by my expected completion date (uh...better make that "dates").

Brenden Tennant eagerly took on the responsibility of getting me out of bed and off to work in the morning.

Kristin Tennant got me out of school.

I appreciate the help that these people have given me throughout my graduate school career. But much more than that, I value their friendship.

This work was sponsored by the Office of Naval Research and Texas Instruments, Incorporated.

TABLE OF CONTENTS

# Chapter 1

# INTRODUCTION

There is a nearly complete absence of meaningful evaluation in current natural language processing research. Forty two papers related to natural language processing were presented at the International Joint Conference on Artificial Intelligence (IJCAI) in 1977. Of these, only four papers described any attempt to evaluate the work being presented other than by giving a few examples of correctly analyzed language. Only one paper [Mann, Moore and Levin, 1977] dealt with the problem of assessing the performance of a model of knowledge understanding. The authors had developed a model for recognizing context shifts in human dialog, and tested it by comparing the performance of the model to human judgements

on the same text. None of the papers that described implemented understanding programs reported on their performance as language understanders. In the other 38 papers, there were no examples of language that was handled inappropriately.

At the next meeting of IJCAI in 1979, the situation had hardly changed at all. There were forty papers on natural language. Two dealt explicitly with evaluation of performance, but since I authored or coauthored them both, we will consider the remaining thirty eight. Of these, three come closest to evaluation of performance. One [DeJong, 1979] that reads news stories off of a wire service reported that about 10% of the stories were understood. The stories were skimmed for the most significant points. No examples were given of successes vs. failures. Another one [Berwick, 1979] that learns syntax rules from example sentences was able to acquire

> "about 70% of a 'core grammar' of English originally developed for the Marcus parser, as well as some new rules...On the other hand, rules for parsing the complicated complement structure of English have yet to be learned, nor is it clear how they might be."

Finally, Harris [1979] described four types of difficult sentences that the ROBOT system has encountered in 12 commercial applications. Thus, 35 of the 38 papers made no mention of evaluation at all, and two of the three that did

made little more than a casual approximation of the success rate. The readers of these 38 papers have very little hope of thoroughly understanding the capabilities of the systems and the techniques described.

Common practice in describing natural language processors is to describe the programs, then give about 20 examples of correctly analyzed sentences. No failures are given. No mention is made of the source of the sentences, whether from the development team or from actual users. No mention is made of how easily a user could express himself within the limits of coverage of the system. In short, the interested reader cannot make a decision based on performance as to which approaches are most promising for further research and development.

The lack of evaluation leaves crucial questions unanswered: what has been accomplished, what problems remain, how general is the solution, and how does it compare with other solutions to the same problem. Generally, comparisons between competing approaches are made on the basis of comparing techniques, not performance. This situation fosters confusion about the progress and achievements of natural language processing research both inside and outside the research community.

> "Proponents of natural language systems cite the success of prototype systems and claim the time is ripe to construct large practical natural language systems. However, those who oppose such systems claim that our current knowledge cannot support an undertaking of such difficulty. A perusal of the natural language question-answering literature indicates the reason for these contradictory claims...With the single exception of the [LUNAR] system there have been no attempts to evaluate the capacity of a natural language question-answering system to satisfy the needs of the user community for which the system was designed. Furthermore, there have been few attempts to characterize the extent of the syntactic and semantic coverage of English provided." [Petrick, 1976]

The lack of evaluation impedes the development of the techniques of natural language processing by leaving readers uncertain about what has been accomplished as opposed to what has been speculated.

> "This muddle finally hurts those following in the researcher's path. Long after he has his PhD or tenure, inquiring students will be put off by the document he has left behind. He seems to have solved everything already, so the report says, yet there is no tangible evidence of it besides the report itself. No one really wants to take up the problem again, even though the original research is a partial success or even a failure! If a student decides [the researcher's idea] is a good idea, and wants to study it, people will assume he is 'merely implementing' an already fully designed program." [Mc Dermott, 1976]

The solution to these problems must lie in more thorough and critical descriptions of the systems.

The problem is that the development of natural language processing technology is proceeding with the benefit of very little data. There is scarcely any performance data in the technical literature that can serve as feedback on natural language processor designs. The most common forms of performance description are sample dialogs and sets of acceptable sentences or texts. These are selected by the authors to depict the extent of competence of the system, rather than to provide a measure of the system's performance. There are generally no examples of failures. There is generally no effort made to describe what is beyond the system's capabilities. The dialog or examples may be typical of the set of acceptable inputs to the system. However, the processing of these acceptable inputs are not typical of the processing of inputs from untrained users. Thus, they do not mark incremental progress. They do not establish a new, clear level of achievement that other research can strive to surpass.

Most of the discussion of natural language systems is more abstract, rather than related to performance. This, in some respects, is appropriate. Natural language processing is a very complex phenomenon. It requires the simultaneous use of a large amount of knowledge. The danger with performance testing is that it has the potential of blurring the most interesting details of the natural language processor by reduction to a simple measure such as the success rate.

Abstract analysis allows more specific issues to be examined in greater detail. It allows some issues to be discussed that relate to a formalism, but not necessarily to an implementation of that formalism in a specific domain of discourse (for example, how grammatical regularities have been factored out, or the ease of implementation in new domains).

Abstract analysis relies on identifying a taxonomy of phenomena that a natural language processor would be expected to handle, then describing how the particular system fares in these categories. The first objection the this technique is that, as practiced, it is unsystematic and incomplete. Authors tend to only present a taxonomy of those phenomena that the system performs well on. The system's performance on the rest of the tasks is left to the imagination of the reader. The second objection is simultaneously an asset of abstract analysis. Incremental progress can be marked by greater articulation of the taxonomy of phenomena and a system's performance within it. For example, say that system A claims to handle some pronouns. Then system B identifies four classes of pronominalization, and can handle all cases in three of these classes. Progress is marked in two ways. First, the taxonomy demonstrates a more detailed understanding of the structure of the problem. Second, the complete coverage of the subclasses indicates a closed area of interest. There are related objections, however. An inarticulated taxonomy does not necessarily indicate

incomplete performance. It may be the case that system A provided more thorough coverage of pronouns than system B. Classes of phenomena are rarely (if ever) claimed to be covered in their entirety. Thus, one would more often see system B partially covering all four subclasses of pronouns, rather than covering any completely. In fact, it is very difficult to establish what "complete coverage" would be. Last, there could be a new system C with an entirely new taxonomy for pronouns. It is generally very difficult to be confident of attained achievement of system C vs system B. The performance of system C may be superior to that of B, but it is hard to make a convincing case for this in a abstract analysis.

Several important aspects of natural language processors are not demonstrated by the current description techniques. One would expect meaningful evaluations to address these issues.

1. A natural language processor can be built, based on the formalism under test, X, to provide coverage for a domain of discourse; furthermore, the coverage would, in some sense, be complete.

2. A natural language processor could be built for a domain of discourse and, though not providing complete coverage, it could provide a subset of English in which the users may express themselves comfortably.

3. A natural language processor based on formalism X would provide superior service to users than one based on formalism Y.

4. A formalism can be readily applied to a number of different domains of discourse.

5. A formalism does not contain holes, inconsistencies or limitations that were unforeseen at design time.

These are clearly goals that are included in the aspirations of natural language research. However, they are not goals that have been demonstrably attained. Significantly, incremental progress toward these goals has not been clearly marked.

If the five points above are, in fact, goals for the development of the technology of natural language processing, they illustrate another facet of the problem. The goals are all very vague. They refer to the system being "complete," "comfortable," "superior," and "readily applied." These terms are vague. If the progress toward these goals is to be marked in any meaningful way, one must be much more specific than this. One of the contributions of this thesis is to express these goals in a more explicit manner, and in such a way that incremental progress can be recognized.

# 1.0 AN EVALUATION METHOD

This thesis will present a method for evaluating natural language processors. It is designed primarily for evaluating user interactive systems, but I feel that the ideas could be extended to include text processing systems as well.

The natural language processor is considered from three points of view:

Habitability Analysis

Completeness Analysis

Abstract Analysis.

Each of these will be discussed briefly below.

## 1.1 Habitability Analysis

The purpose of habitability analysis is to ascertain how well the natural language processor, implemented for a particular domain, performs the tasks it was designed to perform. Data is collected by recording actual transactions with test subjects. The subjects should be familiar with the domain of discourse, but should not be told of any implementational details of the system such as the structure of the database or any limitations of the coverage of the

system. The test problems given to the user to solve with the system should be selected to be within the limits of coverage of the system. The difficulty of the test problems will be graded. The success rate of the system will then be computed. Habitability analysis then, is based on the performance of the system, given problems within the system's capabilities and given users with no prior knowledge of how to use the system.

## 1.2 Completeness Analysis

Habitability analysis examines how well the system accomplishes what is was designed to do. Completeness analysis is intended to ascertain if the system was in fact designed to do the things that users expect it to do. Completeness analysis uses a simulation of a natural language processor. Users who are prepared in the same way as those for habitability testing are given a challenging problem to solve. They are informed that they will be communicating with a researcher through a terminal. The researcher will play the part of the natural language processor, and generate responses to their utterances. The linguistic and conceptual content of the user's utterances are then described. These indicate what the user would like a natural language processor to do. The descriptions of the users' utterances are compared to the capabilities of the natural language processor. In this way, the evaluator can establish a correspondence between what the

system is designed to do and the users' hopes and expectations for it.

## 1.3 Abstract Analysis

A number of significant issues are not directly addressed in habitability and completeness analysis. Among these are approaches toward closure, handling non-determinism, handling incomplete understanding of user's utterances, portability, and underlying assumptions about the user. A taxonomy of considerations has been developed to guide an evaluator to analyze a broad range of issues relevant to natural language processing.

# Chapter 2

## CURRENT EVALUATION PRACTICE

As noted in the previous chapter, little has been done toward evaluating natural language processing systems. In this chapter, we shall discuss the ways in which the performance of systems has been described. Not surprisingly, there is little in the way of reasoned justification for the techniques of performance description that have been applied. There seems to be general agreement that techniques are inadequate, but simultaneous agreement that there is currently nothing better available.

I will describe the following performance description techniques: positive examples, near-miss examples, sample dialogs, hands-on experience and user testing, abstract

evaluation and simulation.

## 1.0 POSITIVE EXAMPLES

Many papers on natural language processors present a number of examples that are handled appropriately by the system ([Woods, Kaplan and Nash-Webber, 1972], [Brown, Burton and Bell, 1974], [Hendrix, Sacerdoti, Sagalowicz and Slocum, 1977], [Waltz, 1978]). The reader is to assume that these examples are in some way typical of the class of acceptable inputs. It is very difficult, however, to reliably extrapolate from the positive examples to the general characteristics of the class.

To illustrate the difficulty, consider the following examples from PLANES.

1) How many NOR hours did plane 3 have in June '73
2) How many hours did plane 3 have of NOR
3) Give me the hours of NOR that it had in June '73
4) What types of aircraft are there
5) What planes do you know about
6) Let my favorite plane mean plane 7
8) Now I want to talk about A7's
9) How many maintenances were performed between May 1 and 7, 1970
10) How many flights did plane 3 perform in May
11) Give me the NOR hours and number of flights for planes 2,3,4
12) Give me the NOR hours for plane 3 for March 2 and 3, 1973.

Among these twelve examples, five were interpreted correctly, and the other seven were interpreted incorrectly. Study them and try to guess which are which. The correctly handled inputs are 1, 4, 6, 9 and 11. The remainder failed for a variety of reasons, some insignificant and others reaching to the very heart of the formalism. The most insignificant failure is in 3, the phrase "hours of NOR," which should have been recognized as a semantic constituent, but was left out of the constituent parser simply by oversight (more on the workings of PLANES later). It would be a fairly simple matter to add it. Example 4 elicits a canned response. Example 5 could as well, but it was not included. More problematic are the facts that noun phrases must be contiguous (ex. 2, "hours...of NOR"). Also, with a few specific exceptions (ex. 4 and 6) all utterances are assumed to be database queries and are interpreted as such (hence ex. 8 fails). Further, semantic interpretations are assigned locally to semantic constituents. General words like "perform" are given domain specific interpretations, assuming that the restricted domain will justify this. "Perform" is given the interpretation "MAINTENANCE ACTION" in example 9 (its most frequent meaning), but this causes problems is example 10. Finally, PLANES can accept a number of types of constructions involving conjunctions (ex. 11), but cannot represent the concept of a list of dates (ex. 12). However, a time spanning between two dates is acceptable (ex. 9).

The five acceptable sentences above are examples presented in Waltz [1978]. They are indeed typical of the sentences that are acceptable to PLANES. As the negative examples show, however, there are other sentences that do not appear to differ dramatically from the typically acceptable ones, but which are unacceptable.

Information on the performance of PLANES was available to me. Information on the performance of other systems was not so readily available, but I believe that PLANES is not an unusual example. Consider two others from different systems. Miller, Hershman and Kelly [1978] point out that four of the following seven examples were acceptable to LADDER [Sacerdoti, 1977]:

13) What is the distance between PECOS and Honolulu
14) What is distance between PECOS and Honolulu
15) What is distance from PECOS to Honolulu
16) What is the distance from PECOS to Honolulu
17) How far is PECOS from Honolulu
18) How far is PECOS from here
19) What is distance from PECOS to here

The acceptable queries are 13, 16, 17, 18. The others were unacceptable because LADDER required "the" before "distance." Note, however, that it was not required before "PECOS." This particular bug would have been fairly straightforward to fix in LADDER. Example 20 was also unacceptable to LADDER, although it is very similar to the acceptable example 13.

20) What is the distance between Honolulu and PECOS.

A third example is given by Petrick who wrote of SHRDLU [Winograd, 1972],

> "[Petrick] presented a list of sentences to T. Winograd, developer of SHRDLU, to determine whether they could be successfully processed. On the basis of our discussion of that list of sentences, the syntactic and semantic coverage provided by SHRDLU appears to be spotty. Although a large number of syntactic constructions occur at least once in sample sentences appearing in published dialogue, our attempts to combine them into different sentences (involving no new words or concepts) produced few sentences that Winograd felt the system could successfully process." [Petrick, 1976]

The technique of giving positive examples clearly does not clarify the limits of performance of systems. As seen from the examples above, although the positive examples may, in fact, be typical (somehow) of the set of acceptable examples, unacceptable examples appear indistinguishable from the acceptable ones.

## 2.0 NEAR-MISS EXAMPLES

I know of only two examples in the literature where utterances that are unacceptable to a natural language processor are presented. One is the evaluation of LADDER [Miller, Hershman and Kelly, 1978] in which 76 utterances

generated from user testing that were unacceptable to LADDER were presented. The same paper presented 25 acceptable utterances. The second example is from Petrick [1976], quoted above, in which he presents 4 utterances unacceptable to NLPQ [Heidorn, 1976].

Near-miss examples can be very useful in clarifying the limits of performance of a natural language system. As we saw in the near miss examples given in the last section, they can at the very least, serve to prevent over-generalization from positive examples. With a sufficient number of positive and near-miss examples, one could hope to get a feeling for the probability that a new sentence would be acceptable.

An extensive list of positive and near miss examples would possibly be useful in understanding the capabilities of a natural language processor, but it may not be concise or systematic. It would also tend to leave the reader with the burden of inferring the rules behind the acceptability or non-acceptability of utterances. Finally, it is a technique that is wholly limited to describing an implementation of a natural language processor in a specific domain of discourse. It says nothing about how well the formalism behind the implementation would fare in another domain. It can not indicate if the near-miss examples were unacceptable due to flaws in the implementation for the particular domain of discourse, or to flaws in the formalism itself.

## 3.0 SAMPLE DIALOGS

Sample dialogs are usually presented as complete dialogs, flawlessly executed, between a natural language processor and a user. They have the advantage over positive examples that they can illustrate the use of cohesive elements such as anaphora. It seems to give the system somewhat more credibility as well if it can be shown to participate intelligently in a prolonged dialog.

The best known system to be described with this technique is SHRDLU [Winograd, 1972]. This system was illustrated with an impressive dialog. The dialog contained a broad array of syntactic structures and an impressive set of conceptual capabilities. But Petrick, quoted in the last section, suggests that the syntactic capabilities were not general. The coverage was "spotty."

Another system that was illustrated with a sample dialog was GUS [Bobrow et al., 1977]. The authors included some discussion of "real" dialogs vs. "realistic" dialogs. The one presented for GUS was "realistic" in that it actually happened, but by a researcher who was well aware of what would and would not be accepted by the system. They warn: "It is much too easy to extrapolate from that conversation a mistaken notion that GUS contained solutions to far more problems than it did." and later, "GUS never reached the stage where it could be turned loose on a completely naive client, however

cooperative." They tempered the possible conclusions from the "realistic" dialog with some difficult examples from a simulated system.

It seems that presenting a flawless sample dialog is ineffective at indicating the limits of performance of a system. It can be useful for indicating, for example, that the system has some pronoun handling capabilities or some ellipsis handling capabilities. These may not be easily demonstrated with a simple list of examples.

A sample dialog complete with some errors, perhaps taken from a test user interaction, may be of more use than a flawless sample dialog. It would be rather difficult to choose one for the purpose of publication. Because of the size constraints on such a dialog, it cannot include a broad range of acceptable and unacceptable inputs, with the enormous variety of potential errors that contemporary systems may have. A few errors in a short dialog can only be suggestive of the scope of the limitations. Also, the reasons for unacceptability may not be clear to the readers of a dialog, and may require extensive explanation. Finally, as with simply listing negative examples, the source of the difficulty may be hard to infer; it may be difficult to tell whether it is due to a limitation of the specific implementation or of the formalism behind the implementation.

Sample dialogs were presented with unacceptable inputs in the description of RENDEZVOUS [Codd et al., 1978]. However, the unacceptable inputs seemed to be presented with an ulterior motive: to demonstrate the use of restricted subdialogs, such as clarification dialogs. The main emphasis of that work was on restricted subdialogs. The natural language component seemed to be a vehicle to get to the subdialogs. Hence, we cannot consider the unacceptable inputs in sample RENDEZVOUS dialogs as attempts to specify the limits of the capabilities of that system.

## 4.0 HANDS-ON EXPERIENCE

In the absence of effective evaluation techniques, some have taken the attitude that the best way to get a good "feel" for the powers of a natural language processor is to sit down and use it. For instance,

> "The only way to accurately assess the level of competence of any natural language processing system is to get actual hands-on experience." [Harris, 1977]

Harris is no doubt mainly concerned with user acceptance -- whether the particular test user would find the system useful. In the present work on evaluation, however, I am primarily concerned with communicating the capabilities of systems to interested readers who are fairly familiar with the

issues of natural language processing. For this audience, a small amount of hands-on experience is of dubious value. First, it can at best represent only a small sampling of the system's capabilities. With the wide range of phenomena that a natural language processor must handle, a small sample cannot provide a thorough test. Second, when failures occur, the user generally has no clues as to the source of the failures, whether these are due to the implementation or the formalism. He also may not be able to determine whether the failure was due to the system's inability to understand the user's phrasing, or whether the user could never get the idea across no matter how it was phrased.

Thus, limited hands-on testing has limitations similar to those for sample dialogs. Hands-on experience does have an obvious advantage over sample dialogs in that the user may try whatever he pleases. This can clarify whether the difference between acceptable and unacceptable inputs is distinct or seemingly random. The user can also attempt to adapt to the limitations of the system. If he can readily adapt to an incomplete system, and if the limitations do not prevent him from doing the tasks he needs to do, then the limited system may be an acceptable alternative to one that performs flawlessly.

Hands-on experience is, of course, an individual experience that cannot te truly shared with others. This makes it generally impractical as a useful evaluation technique. It can, however, be extended to user testing, which will be discussed in the next section.

## 5.0 USER TESTING

An extension of individual hands-on experience is conducting tests with a group of users. A group of users can generate enough sentences so that an accurate profile of the natural language processor can be drawn. A few attempts have been made at user testing, which shall be described below.

### 5.1 The LUNAR Experiment

The LUNAR system was demonstrated at a Lunar Science Conference, where lunar geologists were invited to ask questions of the system. Some question screening was done to eliminate questions containing comparatives and those that exceeded the scope of the database. There were 111 requests entered into the system (by an assigned typist, not by the geologists themselves). Of these, 10% failed due to parsing or semantic interpretation problems. Twelve percent failed due to "trivial clerical errors such as dictionary coding errors which were easily corrected during or immediately after

the demonstration." The remaining 78% were handled satisfactorily . [Woods, Kaplan and Nash-Webber, 1972]

This is an impressive record, but deserves some further consideration. First, the users themselves did not enter the questions, although the requests "were typed into the system exactly as they were asked." Separating the requestor from the terminal would probably have some effect on discourse phenomena, such as use of anaphora. None of the 22 example requests listed in [Woods, Kaplan and Nash-Webber, 1972] contained anaphora. No comment was made on whether the geologists asked series of questions. No comment was made on what instructions the users were given about the system, other than the remark that "many people asked their questions before they could be told what the database contained." Such requests were apparently filtered out by the typist, so the test is primarily one of linguistic coverage as opposed to conceptual coverage. Finally, the presence of a typist and oral requests from the geologists eliminated a number of lexical issues such as punctuation, abbreviation, and spelling correction. For instance, when "K / RB RATIOS" appears in an example request, how was that phrased by the user, "potassium rubidium ratios?" Would "POTASSIUM RUBIDIUM RATIOS," "K RB RATIOS," "K/RB RATIOS" or "K-RB RATIOS" have been acceptable? Again, this informal test examines only the linguistic coverage of the system's syntactic and semantic components.

## 5.2 The LADDER Evaluation

The LADDER evaluation [Miller, Hershman and Kelly, 1978] was a carefully controlled user test of LADDER [Sacerdoti, 1977]. It comes fairly close to my concept of habitability testing. There were 13 users in the test population. The data was lost on one user and two of the 13 users experienced great difficulty with LADDER and were able to make little or no headway with the scenario. Their data were discarded from the analysis. (I find this an insufficient reason to discard data from the analysis.)

### 5.2.1 User Instructions

The users had a significant introduction to LADDER, the test environment and contents of the database. Each user was given a pre-briefing packet with a description of the test situation, the database contents, the information that they would probably be asking for in the course of the test, and a description of the structure of the database, detailing the file structure and names, abbreviations and meanings of the fields in the files. The pre-briefing packets were distributed several days prior to participation in the test. The test users were also given a 90 minute training session immediately prior to the test session. The training consisted of three parts:

1) Tutorial--discussed LADDER'S grammar and the techniques for query entry and editing; examples of LADDER queries were shown including examples of anaphora and "a lengthy discussion of LADDER'S syntax." Use of the terminal was also discussed.

2) Query Entry Practice--the users were required to enter 5 LADDER queries verbatim to acquaint themselves with the keyboard and get further exposure to acceptable LADDER inputs.

3) Query Composition Practice--the test users were given a paper and pencil test of 39 requests for queries (resulting in 390 utterances for 10 users). The users' utterances were then critiqued by a LADDER expert, and five acceptable ways of expressing each request were given to the users.

This amount of pre-test training seems excessive. It becomes difficult to evaluate the central goal that a natural language interaction technology is designed to accomplish--to make a system which is usable without recourse to extensive instruction. In this case, the introduction,

1) illustrated the conceptual coverage of the system by not providing examples for such things as context setting, asking for definitions and reference to discourse objects

2) illustrated the linguistic coverage of the system by providing each user with over 200 examples, and critiques of 39 other requests that he generates

3) illustrated the lexical requirements of the system through examples and feedback.

4) illustrates preferred phrasing through examples

If these test users are to be considered "naive" or "casual" users, it is difficult to determine where their ignorance lies. It is true that they were given 1.5 hours of instruction (plus the pre-briefing packet) rather than, say, 10 hours, but what conclusions can we draw from this?

The point is that providing the user with the details of conceptual and linguistic coverage of the system and providing him with a conceptual model of the structure of the database, largely nullifies the advantage that a natural language processor should be capable of providing. All of the test subjects had had first hand experience with the problem scenario, a search and rescue operation. They consequently had a substantial mental model of what information is needed to conduct such an operation. An ideal natural language processor should permit the user to use his own model of the world until it is found incompatible with the system's model.

Similarly, the users had a command of English prior to the test participation. Why not test LADDER against the user's English rather than test the users' ability to learn LADDER'S subset of English? A test for habitability is a test of the system's ability to accept user utterances without ever (ideally) having to indicate to the user that he has overstepped the bounds of linguistic or conceptual coverage in the system.

A natural language processor is supposed to facilitate a user's task. The user has gained little if he has been given some syntactic freedom, but must still be taught some details of acceptable structures and must learn the details of the structure of the database. If he must know the minutiae of either, he will not be able to use the system effectively on an occasional basis.

> "The LADDER evaluation designers considered the problem of pretest instruction. An intermediate position was adopted with respect to the problem of training. LADDER is sufficiently demanding in its syntax and lexicon so that if no training were provided the outcome of the evaluation could be predicted with certainty--namely, the technology would be severely limited as a tool for accessing a Naval command control database. It was also acknowledged that a higher rate of acceptable queries would have been generated, had the users had more training. However, Such a tactic would have been contrary to the fundamental objective which was to evaluate the LADDER technology 'as is' and with 'fair rules of the game' prevailing -- that is, in a plausible setting with plausible users given moderate training."

I would agree with this point completely. The question is how much training is appropriate. I feel the only answer that can maintain consistency between experiments and imply worthwhile goals for future work is no training in the coverage of the system.

### 5.2.2 Test Problems

The users in the LADDER evaluation were given 15 requests for information that they were to use LADDER to satisfy. The requests were organized in a larger context of a simulated search and rescue mission, but the large context had no real bearing on the test problems. The individual requests were clearly designed to be readily doable by an expert LADDER user. The 15 problems could be solved with 18 queries by an expert.

Most of the test problems were direct retrieval, such as finding the owner and nationality of a ship. Some used the special mathematical functions built into LADDER, which calculate sea distance and travel time. The test problems were presented in written English. To reduce the effect of the wording of the test problems on the users' wording of questions, the test problems were written rather verbosely. Two examples are given below.

> 21) Find the following operational information on the PECOS...
> What is her nationality?
> Then find out her owner.
>
> 22) It appears that there are some readiness problems with those ships.
> Check the database for readiness, reason, casrep, and eicnoms.
> Get this in one list on your display.

Some of the test problems could be entered directly, or nearly

so, such as the examples below.

23) Determine their distances from the PECOS.

24) Which of these ships has a doctor aboard?

25) Also find her port of departure and then her destination port.

I approve the choice of problems used in the LADDER evaluation, but I feel that the character of the problems should have been more explicitly described. There is no particular advantage in giving a user test problems that are considerably beyond the conceptual coverage of the system. We tried this with PLANES. We learned some interesting things from it, but the same lessons could be learned in other, more desirable ways (see the section on user testing in chapter 4).

The LADDER evaluation, then, was centered on the users' ability to use LADDER to solve test problems that they and the experiment designers knew that LADDER could solve. To the degree that we can characterize these problems, we can mark the progress of the field with comparable performance on successively more difficult problems. This approach might be compared to children taking achievement tests in school. A 90% score on problems involving integer addition and subtraction, followed by a 90% score for problems with addition and subtraction of fractions shows progress. If the demands of problems are not characterized in some way, the

scores become meaningless.

## 5.3 The USL Evaluations

USL (User Specialty Languages) [Lehman, 1978] is a system that is based on REL (Rapidly Extendible Language) [Thompson and Thompson, 1975]. The designers have also adopted the REL philosophy that a natural language processor should be a personalized tool for an individual, rather than a standard natural language system made available to a large number of users. Two user studies have been made of USL, and both reflect this view.

The first user study of USL [Lehman, Ott and Zoeppritz, 1978] is an effort to understand what facilities have been omitted from the system that users would have liked to have. It was based on transcripts of dialogs that users had with the system. Failures were noted as possible points for improvement or extension of the system. The familiarity of the users with the system is never described, but it is noted that "the system is not intended for casual users." One must then assume that the users must have been fairly familiar with it. It seems that studying the errors of experienced users for candidates for extension and improvement is perhaps not well motivated. Completeness testing could be better accomplished with a simulated system and inexperienced users. In this way, there would be less chance that the users could

adapt to linguistic or conceptual restrictions (because they wouldn't need to) and they would not approach the system with strong preconceptions about what it could do.

The second user study of USL [Krausse, 1979] was designed in such a way that it described a user's capability to stay within the bounds of the coverage of the system. The study was based on 2200 utterances by one user. This study of the performance of USL with an expert user sheds no light on the main theme of this thesis -- evaluation of the performance of natural language systems with casual users.

## 5.4 Conclusions On User Testing

User testing, like the other performance description methods enumerated above, is a "black box" approach to natural language systems. It only considers input-output behavior, and says nothing about the sources of anomalies. Two factors need close attention if user testing of natural language systems is to stir any interest beyond the bounds of the particular system being evaluated -- 1) the knowledge that the users have about the coverage of the system, both conceptual and linguistic, and 2) the characteristics of the information processing tasks given to the users.

User testing can most profitably be applied to the following question: can casual users effectively employ the system to solve a given class of problems? User testing should not be applied to the study of the characteristics of the users' use of language or their conceptual image of a domain of discourse. Users are capable of changing their habits too rapidly in response to the observed limitations of the system. The change of habits may result in a successful adaptation the system's limitations, or just an unsuccessful corruption of their normal habits of language use. Either way, the data is strongly tainted by the limitations of the system.

## 6.0 TESTING FOR CLOSURE

If a system accepts one particular sentence, there are others that one might assume it should accept as well. For example, if the system accepts example 26, one might reasonably expect it to accept examples 27-33 as well.

26) How many parts did Jones buy from Smith in 1973
27) How many parts did Smith sell to Jones in 1973
28) Did Smith sell any parts to Jones in 1973
29) Did Jones buy any parts from Smith in 1973
30) In 1973, how many parts did Jones buy from Smith
31) How many parts did Jones buy in 1973 from Smith
32) Give me the number of parts that Jones bought

from Smith in 1973

33) Tell me how many parts Jones bought from Smith in 1973

These paraphrases and near paraphrases come very easily to human speakers. One can frequently not remember the exact words he used in a sentence, while he can remember its meaning. This makes user adaptation to a system difficult if the system does not have the capability to accept such sets of similar sentences. The ability to accept such sets is what I call operational closure. To have the ability to accept such sets built into the natural language processor, so that it automatically applies to new concepts or new domains is what I call implementational closure. A system can be evaluated for operational closure through user testing, but implementational closure requires study of the internal workings of the system. Woods was the first to articulate that closure is an important goal of natural language research:

> "The difficulty in natural language understanding is not so much being able to formulate rules for handling phenomena exhibited in a particular dialog, but to do so in such a way that closure is eventually obtained -- i.e., subsequent instances of the same or similar phenomena will not require additional or different rules, but will be handled automatically by generalized rules." [Woods, 1977]

The promise of attaining closure is the attraction to factoring regularities out of the language processing task. If the syntactic facts, for instance, that apply to all sentences can be factored out of the language processing task, centralized, and developed separately, then implementational closure can be gained over syntax. When a new concept is added to the system, the syntactic structures that can be used for describing that concept are ready, waiting and consistent with the syntactic structures available for other concepts. This is one appeal of systems that represent syntactic information separately from other language understanding information. Similarly, it is perhaps the strongest argument against semantic grammars (which do not separate syntactic and semantic information). Concentrating knowledge about concepts into frames and developing conceptual inheritance networks contribute to attaining some degree of implementational closure in knowledge representation.

## 7.0 PETRICK'S TECHNIQUE

Petrick [1976] described an attempt (discussed above) at a comparative evaluation of four natural language processors by submitting to them syntactic variants of sentences that would not substantially change the meaning of the sentences. The syntactic variants were selected from the syntactic structures of other sentences that the systems were reported

to have accepted.

Petrick's technique is one for examining operational closure over syntax. The idea is a valuable one for better understanding the coverage of systems. However, there are three faults with Petrick's description. First, he included no taxonomy of variants. If the experiment were repeated, the experimenter would have to devise his own. Second, he examined only operational closure, with no consideration of implementational closure, in that only the input/output behavior of the systems were considered. Third, he considered only closure over syntax.

The evaluation technique presented in this thesis considers the question of closure in a thorough and systematic way. Equivalence classes are described (classes of sentences that are essentially equivalent, but are not identical) as are perturbation classes (classes of sentences that are similar but not essentially equivalent).

Some researchers in natural language understanding might argue against testing for closure. This might be construed as "linguistic sharpshooting" or an attempt to refute through counter-examples. They could suggest, and with justification, that not all constructions occur with equal frequency in real dialogs, so the failure to accept a syntactic variant of an acceptable sentence is insignificant. The significant measure of coverage is whether the system accepts sentences as entered

by users who are interested in using the system. Sentences entered by researchers who are out to defeat the system (i.e., identify its limits of coverage) are of less interest. This is an opinion one might hear in defense of semantic grammar systems.

It would seem difficult to argue against that idea. If the users can get what they ask for, the system is as good as it needs to be. Perhaps this is true, but it is also true that it has not been demonstrated that current natural language technology has produced such systems. I believe, and I feel most researchers would agree with me, that the lack of a demonstration of such a level of achievement is because it has not, in fact, been achieved yet. What we find are systems whose acceptable inputs are virtually indistinguishable from their unacceptable inputs, as was illustrated early in this chapter. In addition, we are searching for formalisms that can be implemented in different domains of discourse. The domain of discourse has a substantial effect on the nature of the language that users use. For example, it was reported that relative clauses were used very frequently in the LUNAR domain [Woods, Kaplan and Nash-Webber, 1972], but they were used quite infrequently in the PLANES domain. If one formalism were to be transported between these two domains, the absence of relative clauses might not worry PLANES users, but could seriously distress LUNAR users.

## 8.0 SIMULATION

One attempt at a simulated natural language understanding system has been extensively described [Malhotra, 1975], [Malhotra, 1977]. The motivation for this study was to study conceptual and linguistic demands that users would like to make on a natural language management information system. The test users were, in reality, communicating with an experimenter through a computer, while the users thought that they were interacting with a program. The experimenter represented a "perfect" system. This study was done to describe the requirements for a hypothetical management information system. It was not conducted to evaluate an existing management information system.

A second reference [Bobrow, et al, 1976] is made to simulation as an evaluation tool. However, the reference constitutes little more than a mention of having performed the simulation with a few lines of dialog from it.

Simulation can offer advantages for study in evaluation as well as design specification. Simulation dialogs illustrate the conceptual and linguistic coverage that users assume and employ, in circumstances of minimal corruption by language understander limitations. Data can be gathered from simulations that cannot be gathered from interaction with actual systems simply because people can so readily adjust their habits. When a user submits a sentence that is

unacceptable to a system, he will attempt to compensate for what he infers is the problem. He may successfully adapt to the limitation or he may just corrupt his habits of usage to no effect but increased cognitive loading.

A simulated system would exhibit fewer language understanding limitations than an actual system, so the user's language would tend to be less corrupted. It is somewhat misleading, however, to refer to a simulated system as a "perfect" system as Malhotra does. There will be a number of operations that the experimenter-as-query-system will not be able to simulate (like answering, "What were the average incomes of each American over the last ten years?"). He should specify what questions he can and cannot answer. The "perfection" of a "perfect" system lies in its ability to understand what the user has said, not necessarily in the ability to provide the user with every piece of information that might be asked for.

Malhotra's study was primarily exploratory. No pre-test expectations were articulated and no systematic technique was discussed to guide other researchers toward what to look for in simulation dialogs. Thus Malhotra's work is less valuable as a general design or evaluation tool, but it is still valuable as a description of a useful hypothetical management information system.

In the evaluation technique described later in this thesis, simulation plays an important role. There is an attempt, however, to formalize and systematize the use of simulation by presenting an explicit and extensive taxonomy of phenomena to look for in the simulation dialogs.

# Chapter 3

## THE DIFFICULTY WITH THE EVALUATION OF NATURAL LANGUAGE PROCESSORS

I believe that the primary reason that so little evaluation has been done on natural language processors is that it is a difficult task. This chapter will discuss the difficulties. The first section will discuss the goals of an evaluation of a natural language processor, then later sections will discuss a number of the problems that make the task difficult. The next chapter will discuss the specific approach toward evaluation advocated in this thesis.

## 1.0 THE GOALS FOR EVALUATION

The primary purpose of evaluation is to enable comparison. The comparison may be between two systems, between two incarnations of the same system (marking progress), between a system and a set of theoretical limits, or between a system and an independent scale. One may wish an evaluation to measure improvement or superiority or the degree of convergence on a goal.

In order to indicate any of these, an evaluation of a natural language processor must specify the limits of coverage that the system embodies. The limits of coverage have several significant implications for the circumstances under which the formalism may be advantageously applied. These include the match between the formalism and the domain, the match between the formalism and the characteristics of the users and the match between the formalism and the personnel who will be expected to generate and maintain the system in a particular application. The match between the formalism and the personnel to implement and maintain it will not be described further here. The other considerations will be discussed in more detail in the following sections.

## 1.1 Specifying The Limits Of Coverage

The main complication in specifying the limits of coverage of a natural language processor is in confusing an implementation and the formalism behind the implementation. At the current state of the technology, one is primarily interested in formalisms rather than particular implementations. It is far more significant that a formalism has been designed to build natural language interfaces than that a specific interface has been built in a particular domain for a particular set of users. However, the most effective way of thoroughly examining the formalism is to examine an implementation of it in a specific domain. Natural language formalisms are sufficiently complex that if one were to try to understand their capabilities without reference to an implementation, he would soon get lost in the tangle of a large number of interacting facts. It is not particularly useful to state that a natural language processor was capable of handling 65% or 95% of the sentences typed into it. Neither is it particularly helpful to learn that the sentence, "How many hours did plane 3 have of NOR" was misinterpreted by the system, if that were the only information provided on the failure. What must be determined is the cause of each failure. Of the X% that were misinterpreted by the system, how many were due to trivial coding errors (implementational details) and how many were due to limitations of the formalism itself? Of the examples of failures, why did they fail? How

difficult would it be to include the failed sentences within the coverage of the system?

## 1.2 Specifying The System/Domain Match

If the formalism is incomplete, the deficiencies may be important in one domain but not in another. For example, PLANES collects semantic constituents from a sentence, then attempts to interpret the sentence from this set of constituents, but does not use the order in which they occur. This technique caused few problems in the domain of aircraft flight and maintenance records because the relationships between semantic constituents could be determined by the types of constituents themselves. For example, the set (FLIGHTS PLANE-3 DEC-3-1972) is interpreted as the number of flights for plane 3 on December 3, 1972. Regardless of the order of these constituents, one can still infer that within the bounds of the PLANES domain, this is the only plausible interpretation. On the other hand, in a domain of legal interactions, for example, the unordered list (SUED MARY SUSAN) leaves some confusion over who sued whom. The semantic types of MARY and SUSAN are the same, so they give no insight into the directionality of the SUED relationship. In addition, (SUED MARY DEC-3-1972) gives no clue as to whether MARY was the plaintiff or the defendant. This is so in spite of the fact that the semantic constituents are all of

recognizably different types, but the direction of the relationship relies on syntactic information (ordering of constituents) that is lost in a PLANES type formalism.

Many formalisms are written for and tested in one domain only, and as a result may contain many limitations that were insignificant in the implemented domain (and hence went unnoticed) but which could be crucial in a different domain. Part of the problem is that it takes such a large amount of effort to implement a formalism for a domain that the scope of a research project simply does not permit multiple domain implementations. This is another reason why systematic evaluation techniques is so important. They could allow consideration of the potential performance of a formalism in domains that have not been implemented. A systematic examination of important features and issues can yield many of the benefits of multiple implementations without the effort of actually generating multiple implementations.

## 1.3 Specifying System/User Match

Just as limitations may go unnoticed when a formalism is implemented in a limited number of domains, some limitations may go unnoticed when the system has been used by few users, and by users with similar backgrounds. In the extreme, many natural language systems are used only by members of the development group. Others may be used by colleagues who

happen to be walking down the hall and get button-holed into being test users.

People each have their own idiosyncratic ways of expressing themselves, from the lexical level through syntactic habits through conceptual habits and preconceptions. One user that I had use the Automatic Advisor [Tennant, 1977], for example, was deeply entrenched in the use of the term "deal with". At the time, that term was not included in the Automatic Advisor. It kept asking him for paraphrases of his sentences. He was so committed to "deal with" that he would not abandon the phrase even though asked several times for paraphrases. For him, the system was totally unusable simply because it did not include "deal with". This is certainly an extreme example, but it does illustrate that if a system is to be useful to a large fraction of a set of users, it may have to accommodate to idiosyncrasies.

In testing PLANES, the most significant variations between users that were observed involved the assumptions that were made about the concepts that the users referred to. An assumption is built into PLANES that users will, by and large, restrict their utterances to database queries. It was found that this assumption is not always valid. Users referred to concepts that might reasonably be covered in such a database, such as data about pilots or maintenance personnel. However, this data was not in the database and PLANES was not able to

understand enough of what was being asked to indicate that it knew that the information was not available.

## 2.0 CONFUSION OVER SCIENCE VS TECHNOLOGY

Two different goals have often been cited for natural language research. One is that the research would lead to a better understanding of human language understanding and human intelligence. The other is that it would lead to the development of a very useful technology, namely a technology that would enable people to communicate freely with complex and powerful computer systems in a way which they need little or no time to learn, and which they will not tend to forget.

These two goals are blurred by the notion of some that through the development of a technology, we will probably gain insights into the workings of language in humans. One illustration of this is the similarity of results of work on two approaches to parsing. Marcus [1978] designed a parser that would conform to linguistic theories of how humans parsed sentences, particularly with regard to not backtracking in sentences that human speakers do not consider garden path sentences. Simultaneously, Rusty Bobrow [personal communication] was working on improving the efficiency of an ATN parser. He changed the control structure of the parser somewhat, and defined a some new arc types for grouping sets of related arcs. His goal was to improve the parsing speed by

eliminating unnecessary backtracking. In subsequent discussions between Bobrow and Marcus, they found that many of the improvements that each had made had also been made by the other, albeit with different motivations.

The Yale AI group generates a great deal of code, but they claim that very little of it is produced for the purpose of contributing to the technology of natural language processing (there are some exceptions, however, such as the FRUMP program, which is designed to "skim" wire service articles for stories about certain people or events). The purpose of programming in the Yale AI project is to embody a complex theory of language in a form that can be used to test the theory against a set of example texts. They make no claims about the formalism or the implementation of the programs other than to claim that they embody the theory. Once a satisfactory amount of evidence has been accumulated through running the program to give support to the theory, the program is discarded. They claim to be studying cognitive science, not developing a technology.

When the question of evaluation arises, how shall these different approaches be sorted out? It might be tempting to write a program, and if it works well, claim to be developing a technology. If it is too slow and too limited to be useful, claim to be studying a narrow topic in cognitive science (this is often justified -- many of the disappointments in trying to

develop a technology of intelligence stem from underestimating and not fully understanding the complexity of the task).

The programs written in the cognitive science paradigm leave little to be tested. They embody a set of concepts that are known to the system, and which are not claimed to be anything more than the concepts required to run the examples. There are commonly no claims that the programs will run on any other examples without expansion or elaboration of the concepts included in them. There is generally no formalism to examine. The appropriateness of this approach to other domains is not germane. The ability of the system to handle inputs other than the examples without modification of the program is not claimed, so not germane. Assumptions about the types of expected inputs are not germane, since the system is designed to operate on a fixed set of inputs, and no others. Questions of efficiency and closure are not germane, because the program is designed to be discarded after it has been used to evaluate the theory.

## 3.0 HANDLING LARGE NUMBERS OF FACTS

Natural language processors must deal with a large variety of inputs and so must embody and use a large number of facts. These facts include those that are necessary to recognize the range of concepts that users may refer to and the variety of ways in which different users may refer to the

concepts. When it comes time to describe the operation and performance of such a system, it is necessary to indicate in some way whether the system holds the facts necessary for it to do the job it was designed to do. Also, it is important to determine whether it holds the facts, or has the potential to hold the facts, that would be necessary to enable it to do a set of related tasks in different domains or with another class of users with different needs or backgrounds.

One of the contributions of this thesis is to attempt to systematize the facts necessary to do the job of natural language processing. It is an attempt to get a broad brush understanding of what facts are involved in the task, and how thoroughly particular systems incorporate them.

## 4.0 THE NAIVE USER

Although many natural language systems claim to be designed for the naive user, there has not been much discussion in the AI community about who he is or what his characteristics are. The primary assumption is that he is naive about a particular computing system, i.e., he as not learned the nuances of using a particular program, but he has need of the information or services that the program can provide.

## 4.1 Conceptual And Linguistic Flexibility

I feel that the expected characteristics of naive users should be made more explicit when considering natural language processing. The user should indeed be allowed to express himself in the manner to which he is accustomed. This is the familiar sentiment that has been repeated throughout the history of natural language processing. However, the characterization should be taken farther than that. Our experience has shown that users need flexibility not only in allowed linguistic structures, but in concepts as well. An important function of a natural language processor should be to recognize that the user is asking for information or a service that is beyond the system's capability to provide. Users may ask, for example, for data that is not in the database to which a natural language processor interfaces. A system should be able to reply that it has understood the user, but cannot satisfy the request. If current systems cannot satisfy the request, they generally are not prepared to understand what the request means. As a result, if a request for information cannot be understood by a system, it is assumed that this is due to the phrasing of the utterance, and the user gets a message like, "would you please rephrase your query". With this reply, the user would never realize that the difficulty is <u>conceptual</u> rather than <u>linguistic</u>. Perhaps more significantly, he would not clarify his understanding of the system's capabilities through interactive experience.

## 4.2 The User's Misconceptions About The Domain

Another problem that may arise in an interaction is that the user may have misconceptions about the domain itself, rather than simply the system's coverage of the domain. Kaplan [1979] discusses some cases when erroneous domain presuppositions can be detected with database searches. These include problems such as a user asking, "How many of the computer science students who took rhetoric 101 received a grade of B or better?" when in fact, no computer science students took rhetoric 101. These misconceptions can be taken somewhat farther, beyond the range of what can be detected through database searches. For example, there may be a departmental rule that prevents computer science students from taking rhetoric 101. Perhaps computer science majors must take rhetoric 102. Simply stating that there were no computer science students who had credit for rhetoric 101 does not get to the heart of the erroneous presupposition.

A primary advantage of natural language interfaces over other more structured forms of interface is that natural language provides a unique capability to deal with a user's conceptual errors. Enough information is conveyed in language for an intelligent and informed listener to be able to detect conceptual problems and try to correct them. If one eavesdrops on an information gathering conversation, he finds that much of the conversation is devoted to clarifying mental

models of the world. Only after this is accomplished does the conversation move on to providing the information that fits into the model. Since this is a capability that is unique to natural language processing technology, it is one that should be exploited in the technology.

## 4.3 Problem Solving Strategies

We have discussed naive users' conceptual and linguistic demands upon natural language systems and the possibility of their having misconceptions about the domain. A third problem is understanding the level of problem solving that he will use the system for. As an example, suppose a user was asked to investigate the low productivity of a particular plant [Malhotra, 1975]. He has at his disposal an intelligent system for assistance. Why shouldn't the user simply ask the intelligent system to investigate the low productivity problem? The point is that systems, like people, will have a range of expertise in which they are proficient. The user, if he is to make most effective use of the system, must understand what this range of expertise is and exploit it. If the user asks for capabilities beyond those of the system, he will be attempting to over utilize it. If he asks for capabilities that are below those of the system he will be underutilizing it. So in order to make most effective use of the system, the user must have a pretty good idea of the level

and range of problem solving capabilities embodied in the system. And, as mentioned above, a natural language processor is in a unique position detect under- or over- utilization and attempt to educate the user. (It should be noted that while natural language processing does have this potential, the technology has not yet been developed to exploit it.)

These examples demonstrate that knowledge of the users and their knowledge of the system is important in determining what the system should know and how it should behave. A profile of user characteristics form another group of variables in natural language communication which should be considered when evaluating the systems.

# Chapter 4

# AN EVALUATION METHOD

## 1.0 OVERVIEW

This chapter will describe and argue for a method for the evaluation of natural language processors. It involves considering the systems from three points of view:

> Habitability--test the system with users who are familiar with the domain of discourse, but not with the capabilities of the natural language processor
>
> Completeness--simulate the system with a human intermediary and real world problems to study user expectations toward a linguistically capable system; compare results with the capabilities of the system under test
>
> Abstract Analysis--analyze the system from the designer's point of view; abstract analysis allows

consideration of many pertinent issues that are not addressed in "black box" user testing.

Natural language processing is a technology that requires the integration of a large number of facts and processes. In this chapter they are organized into taxonomies so that the analysis will be explicit and clearly organized, and so that the methods presented can be used by others

Insight into any one of the issues in the taxonomies may be gained from more than one of the three viewpoints. The primary result of the evaluation of a natural language system is how and how well the system addresses the issues. It is less critical whether the evaluator's understanding of an issue derived from one viewpoint or another. Thus the description of the results of the evaluation is primarily organized around the taxonomies of issues rather than the viewpoints.

The following sections will describe first the three points of view, then the taxonomies of issues.

# 2.0 THREE VIEWS OF PERFORMANCE

## 2.1 Habitability

The most obvious way to test a natural language processor is to sit some users down in front of it, and let them use it. Transcripts of the interaction can be collected for later analysis. The analysis would include consideration of how frequently the user's thoughts were successfully conveyed to the natural language understanding system. One might consider this the "bottom line" in performance. Indeed, whether a language understander can be implemented for use in a particular domain of discourse is a fundamentally important question.

To understand the significance of habitability testing it would be useful to consider what is involved. Several issues will be discussed in turn.

### 2.1.1 Assumptions About The Users

When a natural language interface is implemented it must be implemented in a particular domain and, less obviously, for a class of users with a particular set of characteristics. For example, when interfacing to a formal database, decisions are made implicitly as to what conceptual image the user will have of the system's capabilities. Consequently, the system will have a built in model (perhaps explicit, but usually implicit) of the kinds of questions the user will ask.

Two extremes of the kinds of questions that could be asked in the PLANES domain are illustrated in examples 1 and 2.

1) Give me the first two characters of the WUC field from the M relation for JCN PE400207734.
2) What caused the increase in down time in 1971.

The reference in question 1 to the first two characters of the WUC field indicates that the user wants only the system designation of the unit on the aircraft that required maintenance (e.g., the hydraulic system, electrical system or landing gear system). Question 1 indicates that the user has detailed knowledge of the structure and contents of the database. His question implies that he assumes that the natural language processor can also understand this degree of specific knowledge about the database.

Question 2 illustrates another extreme. In the PLANES database, the answer to this question is not explicitly stored. In fact, an analysis of the problem of high incidence of down time would be a very challenging task. It would involve the distillation of a large amount of data, and the consideration of numerous hypotheses. It would require that conclusions be drawn using vague and possibly ambiguous information.

Both question 1 and 2 are beyond the capabilities of PLANES. There is thus an implicit assumption that such questions simply will not be asked. This assumption is embodied in the fact that if question 1 were asked, the system would first attempt to make a database query out of the utterance. It might even be successful, but might return data that has little or nothing to do with the actual question. It might be incapable of forming a query, and inform the user of this. But the user would not know whether the system was stymied by what he said or how he said it.

A distinction should be drawn between what a natural language processor can understand and what it can do. Given that a natural language interface will not be able to do everything that its users may wish it to do, it might at least be able to understand references to things it cannot do. It could conceivably be able to understand a request, but not be able to comply with it. This conflicts with the philosophy that understanding is indicated by compliance.

Another form of gracefully declining a request would be the case where the natural language processor is confident of its inability to understand an utterance. If question 1 were given to someone who knew nothing of the PLANES database, he would say with confidence that he didn't know what the question meant -- he didn't know what a "WUC field" or an "M relation" or a "JCN" were. Current natural language

technology assumes that the problems come from elsewhere. A current system would assume the unknown phrases are perhaps misspellings of known phrases. In most instances, in fact, if an entire utterance is not understood, the problem cannot be localized to specific phrases.

The point is that it is imprecise to speak of a natural language system that "answers questions about a database." There is a certain range of concepts that a system is designed to understand. It is important that these concepts match the concepts that the users use when thinking about the information in the database. It is also important to consider how the system should react when these assumptions prove inadequate.

Assumptions about Users

- Familiarity with the database
  - organization
  - contents
- Familiarity with the language analysis component
  - linguistic restrictions (e.g., no pronouns)
  - conceptual restrictions
    - level of abstraction (retrieval vs. analysis)
    - reference to discourse objects
- Familiarity with the domain of discourse
  - esoteric vocabulary required (NORMU, RMCNFE, landing code 5)
  - technical vs common definitions (failure)
  - technical level of displayed information
  - difference in models of domain processes
- User familiarization with the system
  - none required
  - learn limitations through trial and error
  - on-line help
  - off-line documentation
  - on-line coach
  - identification of erroneous presuppositions about data and capabilities

Figure 1 Taxonomy of Assumptions about Users

### 2.1.2 Training

Training users for habitability testing is a critical issue, as was discussed in chapter 2. If the testing is to reflect the system's ability to accept the utterances of users who have not been trained on the system, it is obviously important to avoid training the test users. Once again, training relates to the assumptions that are being made about the users.

The view of training proposed in this thesis is that the users should be familiar with the domain of discourse in which they will be operating. They need not be familiar with the operation of the natural language processor. They need not be familiar with any of the details of the organization or contents of the database that they are using. Ideally, they should be able to attend exclusively to their problem or information needs and not carry the cognitive overhead of trying to understand any of the nuances of how to use the system.

This view of training suggests that ideally one should use individuals who are familiar with the domain, and give them no training whatsoever. If users who are familiar with the domain are not available, the test users should be given instruction on the domain of discourse only.

This view toward user preparation has a built-in bias in favor of systems that attempt to interface directly with users who have generated their own need for information, and against the use of intermediaries. For example, it is biased toward accommodating a scientist, engineer or manager who has need of information and attempts to get that information himself, but against a scientist (with domain expertise) asking his secretary (without domain expertise) to use the natural language system to get information. There would very likely be a difference in phrasing between the manager and secretary's requests. In the event of the system's inability to understand the first request, there would probably be a more dramatic difference between the paraphrases generated by the scientist and secretary. The manager's deeper understanding of the domain allows him to paraphrase his request from a greater variety of points of view.

### 2.1.3 Length Of Use

An issue closely related to training is that of how long the users have used the system. The issue is that if a natural language system is used extensively by a user, he will learn the limitations of the system and adapt his language to them. One commonly cited goal of natural language processing is that it makes information accessible to a casual user -- one who has not learned, or who has had time to forget, the

limitations of the system.

Natural language processing has a unique potential for giving casual users rich access to complex systems. This is the advantage of the technology that is most frequently cited. It is not, however, necessarily the only advantage.

With sufficient exposure, casual users become expert users. (That is to say, they become experts in using the natural language processor. It is assumed that even the casual users are fairly expert in the domain of discourse.) Little consideration has been given to how satisfactory a natural language processor is after a user is familiar with its conceptual and linguistic limitations. It is at this point that extensibility becomes a more significant consideration.

The position that was taken in the evaluation of PLANES was to use several casual users, each in relatively brief dialogs (they averaged 29 questions per user.) The orientation of the design of PLANES was for casual users.

In contrast, REL is oriented toward extensive use by a single user who tailors the language understander to his own needs. As a result, one of the main considerations of REL was that it be easily extended by the user (in fact, REL stands for Rapidly Extensible Languages.) REL English provides a core of language processing capabilities, but relies on the user to

construct the domain-specific language processing capabilities. The result could be a system that is customized to the needs of the user who implemented it for his domain, but which may not be particularly useful to other users. An important consideration in such a system is whether the user can extend the system to accept whatever he would like to express, or if there are conceptual or linguistic limitations inherent in the formalism.

### 2.1.4 Characteristics Of The Test Problems

If the results of habitability testing are to be expressed as success rates, one must understand the bias for or against success embodied in the problems that the users are attempting to solve with the system. The characteristics of the problems that users attempt to solve with the system must be understood in order to interpret the results of the tests.

The approach that we took in the preliminary testing of PLANES for this study was to ask subjects who were familiar with the domain to specify what they considered useful and realistic problems. They did this without any knowledge of the workings of PLANES. The intent was to generate problems that were realistic for the domain of discourse, but which were not specifically biased for PLANES. We found that this technique produced user queries that taught us a great deal about what PLANES could not do, but taught us much less about

what PLANES could do. In short, the problems that subjects formulated were too difficult. We could see little advantage in giving users problems that involved data on pilots, for example, when there was no data on pilots in the database. It is interesting to see how a natural language system responds to such requests, but it seems unkind to the users to give them problems that there is no chance of solving on the system.

The problems that were given to habitability test users were generated by someone who was familiar with both the domain of discourse and the language understanding capabilities of PLANES (me). The problems were selected to be solvable with questions that were well within the conceptual coverage of PLANES. They were also chosen to deal with those concepts that had the most complete linguistic coverage. In other words, they were chosen to be relatively easy to solve using PLANES. The problems were expressed in graphical or tabular form to minimize their influence on the users' choice of language.

If habitability tests are built around test problems that are "easy" for the natural language processor to handle, then incremental progress in system capabilities progress is marked not just by success rates in answering questions but also by the characteristics and difficulty of the problems themselves. In one respect this is like comparing scores on tests in

school for different grades. A score of 90% on a test for addition of integers followed by a score of 90% on a test for algebra word problems shows improvement.

The key to a useful natural language system is the ability to refer to a variety of concepts. Any particular concept may be straightforward, but there can be problems with handling a large quantity of concepts and the interference between them. One must discriminate on a finer scale to select from a larger set of similar concepts. Another problem is that of concepts that require specialized linguistic mechanisms such as hypotheses (if...then), comparatives (more...than...), dates and names. Retrieval strategies may be required, as in answering "how many" questions (should one retrieve, count or add?), or in deciding whether the down time for a time period should be retrieved from daily, weekly or monthly summaries, or a combination of the three. Next, there are queries involving inference and simulation, planning and vague specifications. A large measure of the appeal of SHRDLU [Winograd, 1972], for example, was not its conceptual coverage or linguistic coverage, but its ability to handle some questions involving inference and planning.

Summary of Problem Characteristics

1. Variety of Concepts
2. Specialized Linguistic Mechanisms
3. Simple Retrieval
4. Retrieval Strategies
5. Inference and Simulation
6. Planning
7. Vague Specifications

### 2.1.5 Implementation Vs Formalism

Habitability testing is designed to look at the performance of a natural language processor as it is used by casual users. This gives insight into the operation of a fully implemented natural language processor at the current state of the technology of natural language processing, but the performance of fully implemented systems is really of secondary interest. Primary interest is in the formalism that underlies the implementation. The natural language processing system may perform well or poorly when answering questions about the maintenance and flight records of a group of naval aircraft. But most people who are interested in the technology are interested for different applications. It is very important to them, therefore, to know whether the formalism behind an implementation for one domain can be readily applied to other domains. They want to know how the performance will vary between different domains of discourse.

The distinction between formalism and implementation is of key importance, and it is not addressed by calculating success rates of user utterances. For this reason, I recommend that an evaluation include an abstract analysis of the system. I have generated a taxonomy of issues to guide

abstract analysis. It will be discussed more thoroughly below.

## 2.2 Completeness

### 2.2.1 Coverage And Completeness

A natural language processor must have something to talk about. The range of concepts that are built into the natural language system is the CONCEPTUAL COVERAGE of the system. It must also provide for the various ways in which the user will generate his statements and requests to refer to the concepts covered in the system. The set of features or range of linguistic phenomena that have been built into the system to allow for the diversity of users' language is the LINGUISTIC COVERAGE of the system.

The difference between conceptual coverage and linguistic coverage is fairly distinct. Suppose a user has a particular aircraft in mind that he wishes to refer to. Any of the following references could be appropriate in the proper context:

plane 3
serial number 3
the plane with serial number 3
plane number 3
the plane
it
she
the Skyhawk
the other one

the last one I mentioned
the repaired one

The diversity in the form of the reference may be described as syntactic, semantic, pragmatic, or lexical. The means by which the phrases are interpreted as references to that one particular plane are some of the elements that constitute the linguistic coverage of the system. The concept of the plane is one of the elements of the conceptual coverage of the system.

Noun phrases are not the only units that presuppose elements of conceptual coverage. Prepositions can presuppose concepts (e.g., above, behind, inside) as can comparative constructions (e.g., greater than, more flights than). Conjunctions can imply time sequence (e.g., between May 1 and 8), set membership (e.g., planes 4, 5, 6 and 7), or causality (e.g., the engine failed and it crashed), all elements of the conceptual coverage of a system. There are others, this list is representative, not exhaustive.

The designers of a natural language system build a certain conceptual coverage and linguistic coverage into it. The measure of their success is how well the needs of the users of the system have been anticipated.

Users see a natural language question answering system and the database to which it interfaces through the perspective of their own needs and habits. When a particular user approaches a database question answerer he will expect it to include certain concepts, and he will form his utterances in his accustomed way. The degree to which the concepts that are expected by a set of users can actually be found in the system's conceptual coverage is the CONCEPTUAL COMPLETENESS of the natural language processor, with respect to the set of users. Similarly, the degree to which the language of a set of users is appropriately analyzed by the system is the LINGUISTIC COMPLETENESS of the natural language processor with respect to that set of users.

Conceptual completeness, as defined here, is similar to but differs from the definition of completeness given in [Woods, Kaplan and Nash-Webber, 1972]. They defined a system as "logically complete if there is a way to express any request which it is logically possible to answer from the database". Defining conceptual completeness in terms of conceptual coverage has several advantages. First, it extends the range of concepts from those included in the database to the entire domain of discourse. Second, it permits the restriction of the definition from all requests that are logically possible, to all that a set of users find useful. Third, it retains the requirement that there must be at least one way to reference the concepts.

The definition of linguistic completeness is intended to be similar to the definition of fluency in [Woods, Kaplan and Nash-Webber, 1972]. Both consider the variety of ways in which a concept covered by a system can be expressed. The definition presented here differs from theirs in that it is defined with respect to a set of users.

It is illustrative to consider the current description techniques for natural language processors in terms of conceptual and linguistic coverage, and conceptual and linguistic completeness. When a paper describing a natural language system presents twenty or so questions that are appropriately analyzed, the questions include concepts from the system's conceptual coverage, and their forms and features suggest elements of the system's linguistic coverage. Unfortunately, the conceptual and linguistic coverage of the system is not fully specified by what is found in the examples, and one cannot generalize from them. If no claim is made that the examples were in some sense typical of user inputs, nothing can be inferred about conceptual completeness or linguistic completeness. If the paper goes on to explicitly mention phenomena such as ellipsis, pronoun reference, or comparatives, a comment is being made about the elements of linguistic coverage (and, in the case of comparatives, the conceptual coverage of the concept of comparison). Since this is not being related to the needs of a set of users, it says nothing about linguistic completeness.

One could imagine how these elements of linguistic coverage might affect linguistic completeness. However, more must be learned about the language people use when interacting with computers before natural language systems can be engineered to meet specified goals for linguistic and conceptual completeness within an acceptable tolerance for an expected set of users.

### 2.2.2 Completeness Testing

Habitability testing is used to consider whether the system does what it was designed to do. The system will likely have some limitations, and part of habitability testing is to determine how serious these are. Habitability testing considers the user and system, under the condition that the user is continually learning more about the system and adapting to it.

The habitability view of testing does not fully satisfy the criteria for a natural language processor. Part of the motivation for a natural language processor is that it anticipates the needs of the users, and has them provided for in such a way to minimize the user's need to learn the limitations of the system. What we would like to do is to understand what kind of system the user would like to have. This is the motivation for completeness testing.

In completeness testing, the user interacts with a simulated natural language processor whose function is intended to be generally the same as the actual system under test. The simulation is done by routing the user's utterances to a human intermediary. The intermediary interprets the utterances and generates appropriate responses to be returned to the user. The effect is that the user is now interacting with a "system" with very extensive conceptual and linguistic coverage. The user is able to concentrate primarily on his problem. He need not consider the task of trying to learn the limitations of the system that he is using. He is confident that if he expresses himself in a meaningful way that he will be understood.

There are two benefits in addition to freeing the user from learning about the system. The first is that a broader range of problems can be given to the users. In some preliminary testing of PLANES we gave the users some very difficult problems, many of which could not be solved on the system. We were attempting to have them use realistic problems, but PLANES was not prepared to handle many of them. In particular, when users asked questions that were beyond the conceptual coverage of the system, PLANES gave no clue that this was the source of its inability to interpret the utterances. We found that these difficult problems, though realistic, were not very helpful in delimiting the extent of PLANES' coverage. They were useful in identifying concepts

and constructions that were beyond PLANES' coverage, but they were too difficult to indicate much about what was covered. As a result, we restricted the problems for habitability testing to ones that should be doable with the system. We used completeness testing to gain an understanding of how the users would attempt to solve more difficult problems.

Another phenomena that was observed in testing, and which Malhotra [1975] also reported, was the users' reluctance to put demands on the system that they felt might be too taxing. Some PLANES users refrained from using pronouns, elliptical constructions and conjunctions, even though it cost them considerable effort in retyping the redundancies. When asked why, they often replied that they just assumed that the system could not handle such things, and so did not attempt to use them. Malhotra saw the same reluctance, but in his study the natural language processor was simulated by a person (unbeknown to the users). In spite of the considerable powers of understanding that the simulated system exhibited (i.e., the person acting as the system), some users still restrained themselves because they thought they were talking to a computer which would have limited understanding capabilities.

In our completeness testing, we would like to understand what demands users would make on a system if they felt that they were indeed communicating with an intelligent agent. To assure this, and to avoid the problems of a priori bias that

Malhotra observed (in spite of capable performance), we informed the completeness test users that they would be communicating with a person. The person would interpret their utterances and, using a database, generate responses for them.

We also found it useful to maintain some "distance" between the user and the intermediary. When they were on a too friendly basis, there tended to be a lot of extraneous smalltalk. We wanted to minimize this since we were simulating a man/machine interaction.

I feel that completeness testing is useful in determining how large the remaining task is, how far is it beyond what can be accomplished by the existing system under test. It allows the really difficult problems to be separated from habitability testing, but keeps the scope of the problem of natural language processing well in focus. Also, because it minimizes the corruption of user preconceptions and adaptation limitations, it can serve as a guide to the relative importance of various conceptual and linguistic facilities.

## 2.3 Abstract Analysis

Testing is useful for grounding an evaluation in reality. It makes many problems immediately visible. User testing is most valuable in understanding the input-output characteristics of system. However, there are many

significant issues that are not revealed through this approach. Examples of these are the applicability of the formalism to other domains, extensibility, and the effort required to implement the system for a new domain. These can better be understood through an abstract analysis of the system. In abstract analysis, attention can be given to the design of the system, rather than just the I/O behavior.

The problem with abstract analysis is that it can tend to be rather haphazard and incomplete. To avoid this, we have systematized the abstract analysis by generating taxonomies of issues for consideration in abstract analysis. These will be discussed in later sections.

## 3.0 TAXONOMIES OF ISSUES

### 3.1 Concepts

The conceptual coverage of natural language processing systems is not often explicitly discussed. Most attention has been given to the linguistic coverage of systems. Linguistic coverage has applicability between many implementations of natural language processors, while much of the conceptual coverage is specific to the particular domain of implementation. However, many of the anomalies found in transcripts of dialogs with natural language processors are due to the user's attempting to refer to concepts that the

system does not understand. Of these, most are indeed domain specific, though some are not.

1) Did the number of mechanics stay the same per year?
2) Where is [the work center] located?

3) Is there a reason that you can tell me why the AWM time in 1971 was so high
4) Was parts supply having an off year, thereby increasing the NORS time?
5) Was there a WUC that showed a marked increase of failures...

6) Can I see that data again, please?
7) Can you determine if there was a high concentration of one type of HOWMAL and/or WUC for the 2 ACTORGS PE3 and AC2? But first I would like to see the above ACTORG information for 1972
8) I don't understand these numbers
9) List the flight time for aircraft 1 through plane 50 for december, 1970
10) Make table for total NOR hours and total flight hours for each of the following planes: 13, 14, 15, 20, 22.

Sentences 1 and 2 are references to concepts that are not included within the conceptual coverage of PLANES, i.e., the PLANES database does not include information about mechanics or the locations of work centers. These problems are quite implementation specific. The next group of examples are also fairly implementation specific, but in a different way. Sentence 3 refers to a high-level request whose execution would take greater powers of inference and problem solving capabilities than PLANES has. It would take a significant investigation to determine the reason for high AWM time (in fact the user was involved in a two hour session to determine the reason for high NOR time, which is a similar problem that

he was not able to solve.) Questions 4 and 5 contain vague references to "an off year" and a "marked increase" that could not be understood.

The most domain independent conceptual anomalies are presented in questions 6 through 10. "Again" in example 6 is a reference to the time sequence of the dialog process. Similarly, "but first" in example 7 refers to the time sequence of the dialog process, and its correspondence to the sequence of requests from the user. "These numbers" in example 8 is a reference to the particular representation of the information presented to the user. The definite noun phrase was referring to an aspect of the answer, an object created in the discourse, not to the object referred to in a previous query (as would be assumed by most referent resolution programs). In example 9, the phrase "aircraft 1 through plane 50" assumes that the natural language system understands that the numbers 1 and 50 correspond to an ordered sequence. Each member of the sequence is implied. Finally, "table" in example 10 is a reference to an object to be created for the discourse, not one that is inherent in the domain of aircraft flight and maintenance records.

Examples 6 through 10 illustrate concepts that would be common to many domains of discourse, not just that of navy flight and maintenance records. The preceding examples may not be specifically applicable to other domains of discourse,

but certainly the classes of vague and high level references are germane to all domains. Also, all domains are likely to have to face the problem of what Codd et al. [1978] calls semantic overshoot, the reference to concepts that could reasonably be included within the domain of discourse of the system, but which were not included, such as the references to mechanics and locations.

One natural language system that has given some attention to conceptual coverage is REL [Thompson and Thompson, 1976]. This system has been designed to be applicable to a wide variety of domain applications, and to ease the production of a domain implementation. The formalism includes, in addition to a syntactic grammar with associated semantic rules, some built in concepts such as the ability to handle time relationships (and deduce them from tense information), the ability to handle some arithmetic capabilities (add, subtract, multiply, divide, average), and some database manipulation functions such as image functions (involving displaying data in a partitioned hierarchical form and being able to evaluate functions ranging over the partitions.)

As mentioned above, Codd's [Codd, et al., 1978] formalism gave some attention to conceptual coverage in the form of detecting semantic overshoot. The formalism allowed for the inclusion of keywords that, if found, were immediately taken to mean that the conceptual coverage of the system had been

exceeded. For example, in the parts and suppliers domain in which RENDEZVOUS was implemented, any mention of the word "manager" indicated semantic overshoot.

Work on knowledge representation is also closely related to conceptual coverage. Implicit in much of the knowledge representation work is a set of concepts that should be representable. One example of this is the ability to represent quantification. Another is the representation of time relationships. A third is the representation of causal relationships, and a fourth is belief systems.

### 3.1.1 The Problem With Conceptual Coverage

The main difficulty with considering conceptual coverage is that it quickly becomes domain specific. Within a domain, the conceptual coverage of a system is potentially infinitely extensible. As a result, much of what would be included in the conceptual coverage of a system would have little general significance. Also, with sufficiently narrow domains of discourse, such as those currently being implemented for natural language processors, there is not much that is common to all domains.

I do feel, however, that conceptual coverage is important for consideration in the evaluation of natural language processors. First, the inability to refer to certain concepts

may be inconsequential in a particular domain of discourse, but may be fundamental to a different domain. If one goal of natural language research is to build formalisms that are applicable to many domains, consideration of conceptual coverage is important in determining what concepts and hence which domains are precluded for a formalism. Second, there are a number of concepts that are common to all domains, such as references to objects defined by the discourse itself. These would include references to sentences, displays, the topic of conversation, or the process implied by an extended dialog. Third, although many concepts are not common to all domains, there are many concepts which are common to many domains. A hierarchy of such concepts, parallel to a hierarchy of domains could be established for a natural language system. When the system is implemented for a specific domain, the branches in the hierarchy that are relevant to the domain are selected for inclusion. Fourth, as the developing technology allows the conceptual coverage of natural language systems to expand, one area of expansion will probably be to include more of the most common concepts. This trend would make more of the conceptual coverage of systems common to one another, lending greater justification to including more of the conceptual coverage in the formalism, rather than in each implementation.

### 3.1.2 The Common Concepts

To make a first attempt at identifying the common concepts of English, I have looked toward those concepts for which there has been a considerable allocation of resources. This includes the closed class words, the concepts implied by affixes and inflections, and fairly fixed syntactic structures. Also included are concepts that are common to the situation of man-machine interaction at a terminal, which I assume will be common to most natural language systems for some time.

Common Concepts

- Concepts from closed class words
  - definite reference
  - indefinite reference
  - gender
  - number
    - singular
    - plural (> 1, counted)
    - mass, (non-countable)
  - modality
    - permission (may, can)
    - obligation (must, have to)
    - ability (can, could)
    - possibility (may, can)
    - logical necessity (must, have to)
    - willingness (will, shall)
    - intention (shall, will)
    - insistence (will, shall)
    - prediction (would, will)
    - hypothesis (would, should)
    - tentative condition (should)
    - characteristic activity (would)
    - past state or habit (used to)
  - place
    - position (at)
    - negative position (away from)
    - relative position (by, over, under)
    - path (over, across)
    - direction (up, left)
    - area (in, on)
  - time
    - past, present, future
    - time point (at, on)
    - time span (during, for)
    - relative time (before, after, by, while)
  - causality
    - cause
    - effect
    - enablement
  - purpose (for)
  - recipient, goal, target (for, to, at)
  - source, origin (from)
  - manner (with, like)
  - means, instrument (by, with, without)
  - stimulus (at)
  - accompaniment (with)
  - support, opposition (for, with, against)
  - possession
  - subject matter (about, on)
  - ingredient, material (with, of, out of)
  - sequence (planes 1 through 10)

Figure 2 Taxonomy of Common Concepts

- Domain specific concepts
  - calculations
    - procedure names
    - argument names
    - preconditions
    - results, side effects, partial results
    - interpretation, significance of results
  - database elements
    - data values
    - fields, attributes
    - files, relations
    - virtual fields, relations, files
  - knowledge about domain
    - definitions of terms
    - models of the dynamics of the domain
    - knowledge about different viewpoints
  - knowledge about system
    - calculating capabilities (Can you do percentages?)
    - extent of the domain of discourse (Do you know about pilots?)
- Hypothesis testing
  - models of the effect of changes (Would profits have increased if we had had smaller inventories?)
  - reasonable assumptions
- Logical propositions
  - negation
  - disjunction
  - conjunction
  - quantification: universal and existential
- Quantitative propositions
  - numerical quantifiers
  - comparatives
- Numerical functions
  - arithmetic functions
  - set operations
  - ordinality
  - cardinality
  - conceptually clustered function sets (e.g., application calculators)
- Concept association
  - individual --> event participation
  - individual --> class membership(s) --> other members
  - class <--> prototypical member
  - individual event --> generic event (case replacement)
- Discourse objects
  - sentences (Does <u>that</u> mean that I must take CS 101 before 102?)
  - topic of conversation (the last plane I asked about)

Figure 2 (cont.) Taxonomy of Common Concepts

- future context (context setting)
- output format (give me a table of ...)(express sales in millions)
- the process implied by a dialog (now do all that again for plane 3)
- system generated displays (that table)
- motives and goals of the conversants

Computer dialog objects

- the terminal and its parts
- the computer and its parts, function, status, monitor, operator, etc.

Extensions

- equivalent terms and abbreviations (JFK for Kennedy, crone for female who is more than 85 years old)
- new concepts
  - named subsets/supersets
  - named processes, objects, states, calculations, etc.
  - integration into knowledge base
    - synonyms
    - antonyms
    - class membership and inheritance
    - intersecting/exclusive classes
  - limitation to possible new concepts
- altering a formal database
  - updating contents
  - virtual fields
  - virtual relations

Figure 2 (cont.) Taxonomy of Common Concepts

## 4.0 LINGUISTIC FACILITIES

The linguistic coverage of a natural language processor is a description of the ways in which concepts can be expressed. Much of the linguistic coverage can be built into a formalism in a more domain independent fashion than can the conceptual coverage. Consequently, linguistic facilities have attracted most of the attention in natural language processing research.

The most highly developed aspect of linguistic facilities in natural language systems is syntactic analysis. Behind this work lies a considerable body of theory, much of which has been transplanted directly from linguistics to natural language processing research. Semantic analysis is much less well developed than syntactic analysis. Its domain dependent nature has discouraged the development of general theories. However, recent work has made considerable progress in the area of semantics, particularly with respect to problems of reference [Grosz, 1977], [Sidner, 1979]. One area that is clearly in need of advances in the state of the technology is semantic interpretation. There currently exists no semantic interpretation technique which is simultaneously fast, efficient and attains a high degree of closure over semantic analysis. The semantic interpretation schemes that are currently in use do not do an adequate job of factoring out regularities and generalizing the semantic rules.

In perusing the taxonomy of linguistic facilities presented below one finds that it is a fairly long list and that the elements of the list do not all have an equal probability of occurring. There are several interesting considerations about the relative importance of various linguistic facilities.

First, there is a relationship between the domain of discourse and the language expected from the users. Consider the moon rocks analysis domain for which LUNAR was implemented. Most of the objects that one would want to refer to do not have familiar names. As a result, it was reported that relative clauses were used extensively to identify objects by description rather than by name [Woods, Kaplan and Nash-Webber, 1972]. This domain was also devoid of semantically rich verbs. The model that users were assumed to have was that the system would only retrieve information about the characteristics of the lunar samples. The verb used nearly exclusively for these requests was "be." As a result, one does not find many adverbs, adverbial phrases or participles. Another domain feature that can affect language use is the presence of numerical information. If it is present, calculating capabilities such as adding, averaging and comparing are likely to be referenced by the users. They would not be needed in the absence of numerical information (except, perhaps, for the ability to count).

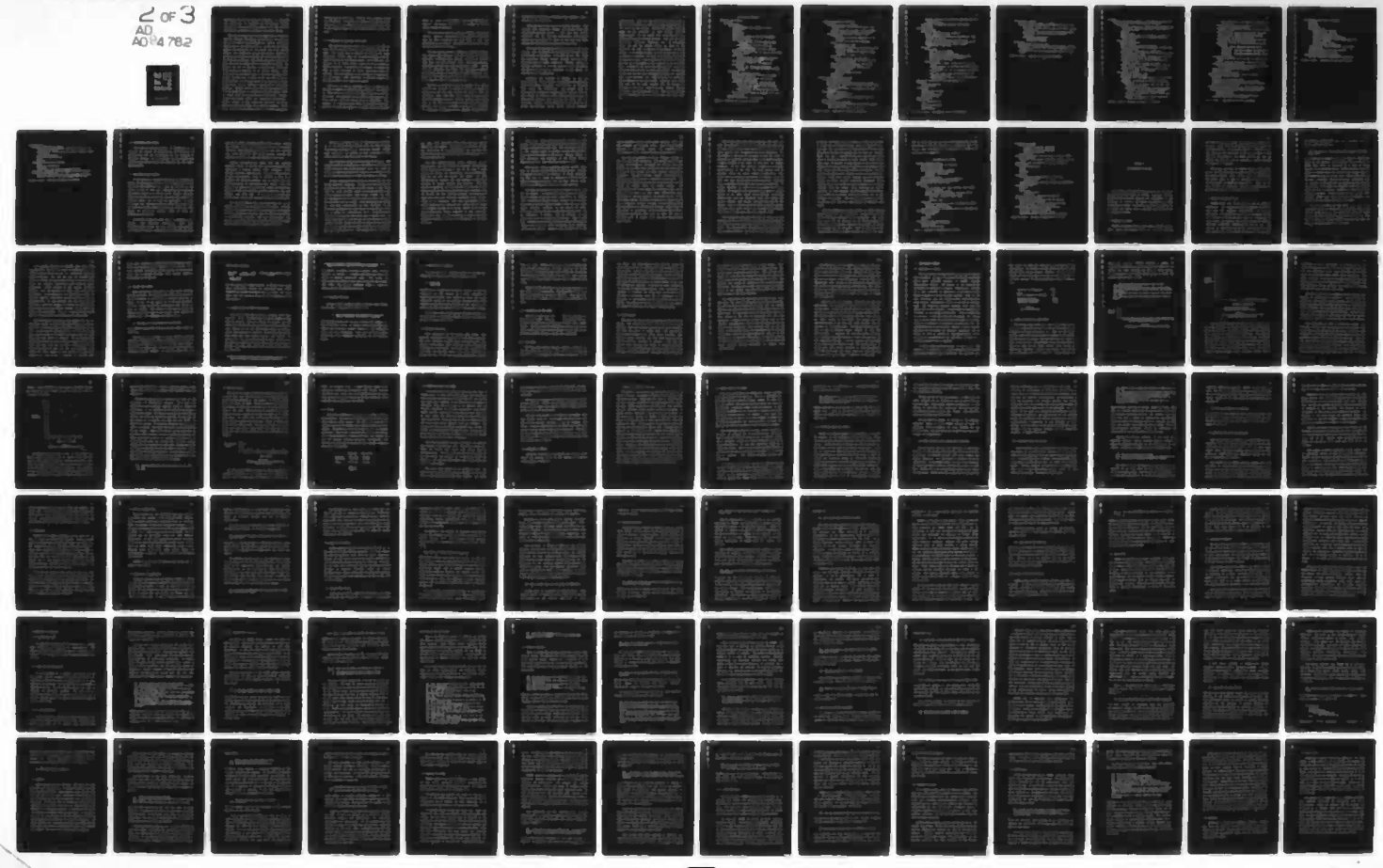
The second factor of the relative importance of linguistic facilities relates to variability among users. A large sample of users would produce dialogs with a wide range of linguistic phenomena. There is much less variability within the utterances produced by a single user [Michaelis, Chapanis, Weeks and Kelly, 1977] (particularly if the system readily understands the user and does not ask for frequent paraphrases.) This observation is a source of divergence of opinion about whom a natural language processor should serve; whether it should serve a single user or a large set of users. Most system designers have made the (implicit) decision to serve a large set of users. To do this a great deal of effort must be expended to include broad linguistic coverage (and broad conceptual coverage) in the system. But the process need be done only once. A single user system, on the other hand, need not include such broad linguistic coverage because the single user's patterns of usage would not require it. Consequently, a great deal of code that would probably never be used by the individual user can be avoided. Such a system would be more streamlined. The most important advantage, however, is that the conceptual coverage would be tailored to the individual's needs. Among the drawbacks of individualized systems are that 1) each one must be implemented separately, 2) they may not be useful to users other than the one they are implemented for, and 3) since language rules and concepts may be highly interdependent, a natural language processor must be

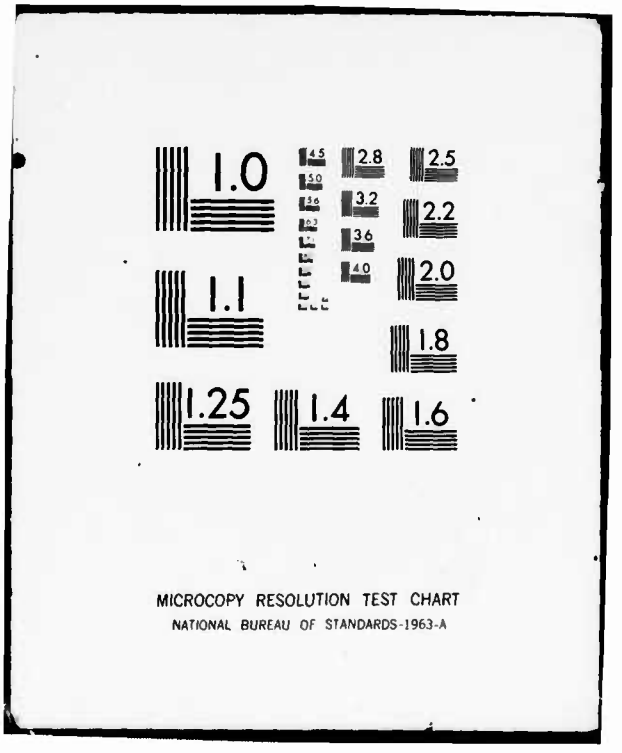
implemented with care and attention to detail; an untrained user attempting to implement his own natural language processor may not be sufficiently familiar with the possible interactions to produce an extensive, operational system.

The third factor of the relative importance of linguistic features concerns limitations of the system. Any system at the current state of the technology will exhibit limitations shortly after a user starts using it. If the user can readily adapt to the limitations of the system and still express himself, the limitations are not very serious. However, it may be quite difficult to identify what the limitations are so that they may be avoided. Most systems provide no clues as to what was or was not understood in an utterance or whether the problem was due to spelling, an unknown word, a difficult construction or a reference to an unknown concept. Also, limitations are not always based on the rules that users might assume. The fact that in PLANES "flight and NOR hours" is misinterpreted is not generalizable to an inability to handle conjunctions. In fact, "NOR and flight hours" is interpreted properly. (PLANES cannot handle ellipsis over conjunctions. "NOR and flight hours" is interpreted correctly because "NOR" is always taken to be equivalent to "NOR hours"; however, "flight" and "flight hours" are interpreted differently.) Another problem is that linguistic facilities may not be consistent over all concepts (as in semantic grammars), which makes inferring a few general rules of limitations impossible

AD-A094 762 ILLINOIS UNIV AT URBANA-CHAMPAIGN COORDINATED SCIENCE LAB F/G 5/2
EVALUATION OF NATURAL LANGUAGE PROCESSORS.(U)
NOV 80 H R TENNANT N00014-75-C-0612
UNCLASSIFIED T-103 NL

2 OF 3
AD
AD-A094 762




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

implemented with care and attention to detail; an untrained user attempting to implement his own natural language processor may not be sufficiently familiar with the possible interactions to produce an extensive, operational system.

The third factor of the relative importance of linguistic features concerns limitations of the system. Any system at the current state of the technology will exhibit limitations shortly after a user starts using it. If the user can readily adapt to the limitations of the system and still express himself, the limitations are not very serious. However, it may be quite difficult to identify what the limitations are so that they may be avoided. Most systems provide no clues as to what was or was not understood in an utterance or whether the problem was due to spelling, an unknown word, a difficult construction or a reference to an unknown concept. Also, limitations are not always based on the rules that users might assume. The fact that in PLANES "flight and NOR hours" is misinterpreted is not generalizable to an inability to handle conjunctions. In fact, "NOR and flight hours" is interpreted properly. (PLANES cannot handle ellipsis over conjunctions. "NOR and flight hours" is interpreted correctly because "NOR" is always taken to be equivalent to "NOR hours"; however, "flight" and "flight hours" are interpreted differently.) Another problem is that linguistic facilities may not be consistent over all concepts (as in semantic grammars), which makes inferring a few general rules of limitations impossible

(because they do not exist). Instead, many specific rules must be learned. (It may be more useful to the user to have a less powerful natural language system that is consistent, than a more sophisticated one that is more difficult to learn to use.)

## 4.1 A Taxonomy Of Linguistic Facilities

The table of linguistic facilities given at the end of this section is organized around facilities that the user may need. The intent was to provide an overview of these facilities. Indeed, most of the section headings could be expanded further to show greater detail. But I feel that the taxonomy as it stands gives a good view of the breadth of common facilities. In presenting the taxonomy, an attempt was made to avoid tailoring it toward a particular theory of language. With the examples, it is intended to be essentially self-explanatory. Some discussion is required, however, of the second portion which deals with closure.

The section on closure represents issues in operational closure (as opposed to implementational closure -- see chapter 3). This section is intended to be used to examine classes of utterances. The acceptance of one sentence would suggest that other similar sentences should also be acceptable. For example, if a system accepts, "What are the hours of NOR for plane 3?" we would also expect it to accept, "What are the NOR

hours for plane 3?" and perhaps, "Give me the NOR hours for plane 3." The closure taxonomy is an attempt to capture thoses classes of sentences.

The first major section is that of semantic equivalence classes. Semantic equivalence classes consist of utterances that all mean essentially the same thing -- they are paraphrases of one another. The variations may be due to word choice or structural variants.

The next section describes semantic perturbation classes. The members have meanings that are similar but slightly different. In PLANES, users commonly referred to aircraft that were "not operationally ready". Some users, however, asked about aircraft that were "operationally ready". PLANES did not understand that term. The advantage of semantic perturbation classes is to examine how users extrapolate from forms they have seen work in a system to other forms that they would expect to work in the system as well.

Next are the syntactic equivalence classes. If a system includes the concepts of numbers of flights and numbers of flight hours, and "How many flights did plane 4 fly" is acceptable, then, "How many flight hours did plane 4 fly" should be acceptable as well. The most striking area where this is not seen is when systems include "extra" conceptual coverage. For example, in PLANES a wide variety of utterances is acceptable for database queries. However, when asking for

the definition of a word, or defining a new synonym, only a few wordings are acceptable.

Following that is the section on spelling variants. This includes both intentional changes such as affix adding and unintentional spelling errors. I feel that this section represents one of the most consistently underutilized regularities in language, in spite of the fact that many systems have some misspelling recognition facilities.

The final section consists of only one point -- that the language that is generated by the system for the user should be acceptable to the system. It is often not the case since the generation (usually "canned" phrases) and recognition facilities have no common components. However, since the language presented to the user could be a guide for leading the user to language that is acceptable to the system, it deserves careful consideration.

Following the taxonomy on closure is one on discrimination. While closure is concerned with seeing the similarity between utterances, discrimination is concerned with seeing the differences. The primary type of discrimination is made on a semantic basis. Here we consider such things as modifiers whose meaning depends on what is modified and synonyms whose meanings are close, but not identical. Another type of discrimination is the ability to detect the different intent behind various syntactic

constructions. Last are the abilities to detect whether a word has been misused or a concept has been referenced that is not known to the system. Consider the example, "How many zones have gas stations" is from Petrick (personal communication). In the domain from which this was taken, parcels of land have a zoning attribute; they can be zoned residential, industrial, commercial, etc. In the example given, the user has misused "zones", at least in the restricted use that is expected in this domain. As another example, in the PLANES database, an entry is made when a part is removed or installed. But there is no explicit record made of replacing a part. A part removed and installed on the same day may be a replacement or just a removal to get to another part. If asked for the "parts that were replaced", should the system use some heuristic to infer replacements? The question implies that the user has a presupposition about the database that is not justified. But a loose definition of replacement may be adequate for the user's needs. Users may also assume that they can use other concepts that are related more distantly to the domain. Some of the test subjects referred to pilots and maintenance personnel. These are not included in the PLANES domain, but it is conceivable that they might be.

Linguistic Elements

- Reference to concepts
  - by name
  - by description
    - noun modifiers
      - predeterminers (all, double, one-third)
      - determiners
      - ordinals
      - quantifiers
      - adjectives
      - identifiers (plane 3)(ice station zebra)
      - relative clauses
      - non-finite clauses
        - -ing participle (The man wearing dark glasses ...)
        - -ed participle (the man fired for incompetence ...)
        - infinitive clauses (the place to eat)
      - prepositional phrases
      - noun-noun modifiers (the jet engine water pump housing screw)
      - participle
        - -ing participle (the crumbling cottage)
        - -ed participle (the bombed village)
      - genitive (the miller's tale)
        - headless (I went to the dentist's)
      - appositives
        - phrases (Dr. Wells, an internist)
        - clauses (the fact that he wrote a letter to her)
    - adverbials
      - adverbs and adverb phrases
      - noun phrases (Peter was playing last week)
      - prepositional phrases
      - finite verb clauses (Peter was playing although he was tired)
      - non-finite verb clauses in which verb is:
        - infinitive (Peter was playing to win)
        - -ing participle (Making a lot of noise, they praised Tom)
        - -ed participle (If urged by our friends, we'll stay)
      - verbless clauses (While in London, we'll stay in a hotel)
    - explanatory text
    - verbal illustrations and examples

Figure 3 Taxonomy of Linguistic Elements

- anaphoric/cataphoric reference
  - pronouns
  - definite noun phrase anaphora
    - superordinate (the a7 with stripes...the plane)
    - general word (the aldermen...those idiots)
    - related concept (my rifle...the trigger)
  - antecedents and referents [Webber, 1978]
    - individuals
      - explicitly mentioned (John went to the circus)
      - implied (John gave Sue a T-shirt)
    - sets
      - enumerated as a set
      - predicated as a set
      - implied as a set by quantification (every dog with spots)
      - implied by association (John gave Sue flowers. They are in love.)
    - stuff
      - specific quantity (my last beer)
      - the stuff in general (My beer was good. But its fattening)
    - generic classes of specific individuals
    - prototypical members of sets
    - events, actions, states, propositions
      - individual events
      - sets of events
      - generic classes of events
    - descriptions
    - predicates
- Wording for emphasis and brevity
  - ellipsis and substitution
    - nominalized adjectives
    - over conjunction (the yellow cars and trucks)
    - substructural (How much is the yellow candy? How much is the red?)
    - superstructural (What is the speed of the JFK? Length?)
  - non-anaphoric omissions
    - reduced relative clauses
    - omitted prepositions
    - omitted determiners (give me planes with > 20 flighthours)
  - abbreviations
  - relying on shared knowledge
    - pragmatic ellipsis (How many flights did plane 3 fly in May. 1971? How many flights did plane 4 fly?)

Figure 3 (cont.) Taxonomy of Linguistic Elements

- assumed arguments for numerical functions (percent downtime)
- analogy, metaphor
- composition of elaborate concepts from simpler sentences

Syntactic forms
- capitalization
- special structures
  - person names (Phineas T. Bluster)
  - dates (December 7, 1941)
  - organization names (The Heart Association, The League of Nations)
  - idioms (kicked the bucket, the thing is)
  - amounts of money ($1.4 billion, 37 cents, $.02)
  - mathematical expressions (gross income - deductions) * tax rate
- sentence
  - basic types
    - SVA (She is in London)
    - SVCs (She is a student)
    - SVO (He heard the explosion)
    - SVOiOd (He offered her some chocolates)
    - SVOCo (They elected him chairman)
    - SV (The train arrived)
  - special types
    - existential "there" (there is a rock in my shoe)
    - cleft sentences (It was John who wore the suit last night)
    - anticipatory "it" (It is a pleasure talking to her)
    - if...then (what would the volume be if we halved the price)
    - comparatives (did plane 3 have more flights than plane 2)

Speech act recognition, indirect speech acts

Negation
- predicate negation
- set exclusion

Conjunction
- clause
- prepositional phrase
- noun phrase
- prenominal modifier
- other structures

Quantification

Punctuation
- .?!, -&+%$=#*^´"@
- <>[]()
- English pound sign
- cent sign
- grave accent

Figure 3 (cont.) Taxonomy of Linguistic Elements

Parenthetical remarks
    explicit function (e.g. ...) (i.e. ...)
    implicit function (Smith, 1980) (the gangster's wife)
        references (Smith, 1970)
        appositives and asides (the gangster's wife) (this is important)
        abbreviation [the Office of Naval Research (ONR)]
        post-nominal modifiers [change in revenues (percent)]
        mathematical expressions
            nested expressions
            procedure arguments
Quotations
    conversations and quoted text
    references to words themselves ("Paris" has 5 letters)
    introducing new words (the "linguistic coverage" of a system is...)
    non-grammatical or non-standard usage (beat it "yous" guys)

Figure 3 (cont.) Taxonomy of Linguistic Elements

Closure
    Semantic equivalence classes -- variations in which the meaning of the unit is essentially unchanged.
        word choice
            naming vs description (What are the prerequisites of CS 240, What courses must I take before I may take CS 240)
            synonyms
            abbreviations
            special symbols ($, %, &, #, @, +, -, =, ?, <, >)
            numbers (one, seven, 1, 7, nineteen seventy, thirty-eight)
        sentence structure
            adverbial location
            passive/active
            single sentence, multiple sentence
            subject-verb inversion (The milkman is here, Here is the milkman)
            declarative, interrogative, imperative
            cleft sentences (It was John who wore argyl socks, John wore argyl socks)
            extraposition (That he is dishonest disturbs me, It disturbs me that he is dishonest)
            subject raising (It is certain that Judy will accept the offer, Judy is certain to accept the offer)
            object raising (It is hard to catch a lion, A lion is hard to catch)
            Presentative there (A snake is in the garden, There is a snake in the garden)
        noun phrase structure
            postnominal modifier order
            prenominal modifier order
            discontinuities (how many hours did plane 3 have of NOR)
            prenominal vs postnominal modifier (preposed adjectives)
                adjective vs rel clause (the red barn, the barn that was red)
                participle vs rel clause (the burning bush, the bush that was burning) (the bombed cities, the cities that were bombed)
                genitive vs preposition (John's house, the house of John)
                genitive vs rel clause (John's house, the house that John lives in)

Figure 3 (cont.) Taxonomy of Linguistic Elements

- nn vs preposition (the fuel tank, the tank for fuel)
- implied vs explicit relationships (the fuel tank, the tank for containing fuel)
- relative, finite clauses (who was riding the bike, riding the bike)
- relative clause, prepositional phrase (the lion that is in the cage, the lion in the cage)
- relative clause, adverbs (the people who are downstairs, the people downstairs)
- fact deletion (We realized that he was lame, We realized the fact that he was lame)
- appositive, infinitive phrase and gerund phrase synonymy
  - gerund: Ray's answering the door was a surprise
  - infinitive: For Ray to answer the door was a surprise
  - appositive: That Ray answered the door was a surprise

verb phrase structure
- detached particles (pick the dog up, pick up the dog)
- dative (gave the key to you, gave you the keys)

prepositional phrase structure
- stranded prepositions (Which house does John live in, In which house does John live)

conjunction and constituent sharing

repeated reference

ellipsis and substitution

Semantic perturbation classes -- units in which the meaning changes slightly
- negation
- determiners and quantification
- antonyms (on a continuous scale: big vs small)
- complementarity (on a binary scale: male vs female)
- point of view (buy vs sell)

Syntactic equivalence classes -- units in which the structure is essentially unchanged, but the meaning is changed syntactic regularity over utterance types
- database queries
- word definitions
- questions about the system's capabilities

syntactic regularity over concepts

Figure 3 (cont.) Taxonomy of Linguistic Elements

- Orthographic perturbation classes
  - prefixes
  - suffixes
  - etymology
  - inflections
  - spelling errors
    - nonsense words (ksdfkj)
    - digraph switch (swithc)
    - stuttering (stutttering)
    - letter replacements (vibeo)
    - phonetic equivalence (shevrolay)
    - terminal dependent errors [(subl (addl x 9)]
    - run-ons (thankyou)
    - extra letters (howxdy)
    - missing letters (telvision)
  - unknown words (aglets, gonocyte, goatsucker, plagioclase)
  - undefined abbreviations and acronyms
  - contractions (didn't, can't)
  - clipping (phone, photo, ack)
- Recognition/generation consistency
  - can all generated utterances be recognized?

Figure 3 (cont.) Taxonomy of Linguistic Elements

Discrimination
- Semantic perturbation classes
  - implicitly graded antonyms (the house is not big <> the house is small)
  - synonyms (get, obtain, procure, secure, acquire, gain, win, earn)
  - word sense selection
  - restrictive modification
  - tense
  - modality
- Syntactic perturbation classes
  - focus
  - new/given information
  - formality
  - restrictive, non-restrictive modification
- Misuse of vocabulary (how many zones have gas stations)
- Erroneous presupposition
  - near miss (how many parts were replaced on plane 3)
  - semantic overshoot (how many pilots flew in june)

Figure 3 (cont.) Taxonomy of Linguistic Elements

# 5.0 IMPLEMENTATIONAL ISSUES

How a natural language processor is implemented is of interest for reasons of understanding its efficiency, its extensibility and the applicability of the formalism to different domains. In this section we will consider some issues of the programs behind the formalism and the domain implementation.

## 5.1 Implementational Closure

In the section on linguistic facilities, we discussed the desire for operational closure -- the situation where all the variations that would occur to a user be accepted by the system if any of them are. One possible way to approach operational closure is to list all of the possible sentences (one could not attain closure this way since there would be an infinite number, but it could be approached.) A more attractive way (and more promising) is for the system itself to embody the necessary generalizations to attain the same degree of operational closure. This is known as implementational closure.

Significant success has been had for implementational closure over syntactic information. Much less implementational closure has been attained over semantic interpretation, however. Generalization hierarchies show some

promise for greater implementational closure [Finin, Goodman and Tennant, 1979], but this has yet to be demonstrated as effective. Some implementational closure has been attained over word recognition with affix analysis programs and spelling correctors, but much work remains in this area.

Another implementational issue that has received attention lately is how to deal with non-determinism and uncertainty in parsing [Marcus, 1978] [Ginsparg, 1979] [Bobrow, personal communication]. This work is motivated by at least three problems. First, in watching a trace of a standard ATN syntactic grammar, one is struck with the surprising amount of backtracking that occurs. With the backtracking, much of the parsing time is spent in reexamining what has already been examined, and often computing the same results, time after time. If parsing could be made more deterministic, less of this wasteful backtracking would occur.

The second problem is that many groups would like to do semantic and syntactic interpretation concurrently. If there is to be both semantic and syntactic processing at incremental points in processing a sentence, then the cost of backtracking is significantly increased. The more deterministic the parsing can be made, the less penalty one is paying for doing semantic analysis concurrently with syntactic analysis.

The third motivation is one of psychological validity. It is the feeling of some [Marcus, 1978] that human language understanders backtrack only in relatively uncommon "garden path" sentences. An example of such a sentence is, "The cotton clothing is made from grows in Mississippi."

One of the main causes of backtracking is ambiguity (and trying to resolve it prematurely.) The problem of ambiguity is so pervasive that many system designers skirt the issue by relying on restricted domains of discourse to eliminate it and on extensive backtracking to handle what ambiguity remains.

Another implementational problem is the accuracy of the system generated responses. Systems could have problems of precision and recall like any retrieval system, or they could have the added problems caused by misunderstanding in the natural language processor. Some systems (PLANES, for one) are implemented with the assumption that the users' utterances are meaningful and well formed. These systems make an effort to give some interpretation to each utterance. If an utterance is anomalous for some reason, the system will still try to assign an interpretation to it. It may be able to "interpret" an utterance that should make no sense at all. It could return data to a user who may not have noticed the error in his utterance. He may then take the data that was returned to him as the data that he meant to ask for, even though it may not be that data at all. Some systems, on the other hand,

are much less forgiving about what they will accept (RENDEZVOUS, for example). This more strict approach reduces the possibility of misinterpretation, but increases the possibility of not interpreting a well-formed utterance or a slightly misformed one.

The next implementational issue, handling partial understanding, has been alluded to elsewhere. When a system cannot interpret an utterance it can do the user a great service by telling him specifically what its problem was. That way the user can correct his query. In many implementations, however, the only response is something on the order of "Please rephrase your question". With the variety of conceptual and linguistic problems that the natural language processor may have encountered, such a response does not provide the user with much guidance. It is painful to watch users trying one contorted paraphrase after another, trying to be understood, when their only problem may be a simple misspelling or a reference to a concept that is beyond the conceptual coverage of the system. Kaplan [1979] has done work on making more helpful responses to user utterances, particularly when the user's utterance is based on erroneous presuppositions about the data in a database for a retrieval operation.

Codd, et al. [1978] gave considerable attention to the problem of how to handle the machine/user dialog when natural language communication breaks down. The idea behind this work is that dialogs should be precise, and when ambiguities or other imprecisions remain in the natural language dialog, a more restricted, less ambiguous (to the system) form of communication (e.g., menu selection) should be used.

Most current question-answering systems present their data to the user through tables or canned phrases. This may be adequate for systems that interface to formal databases, but they will no longer be adequate to interface to more complex knowledge structures. For the more complex tasks, some form of language generation is a strong candidate.

The issue of portability is one that seems to go in and out of fashion. It would clearly be beneficial to create a natural language system that could effortlessly be "plugged into" and existing database system. However, one of the main advantages of natural language systems is that they can apply domain dependent knowledge to the analysis of users' utterances, allowing the user to omit some concepts which can be understood in context. This is difficult (if not impossible) to do in a domain independent way. Some systems have attempted to deliver satisfactory language understanding capabilities to the user while minimizing domain dependent implementation effort. Examples are the ROBOT system (now

called INTELLECT) [Harris, 1977], LIFER [Hendrix, 1977] and REL [Thompson and Thompson, 1975]. Each of these systems requires a significant amount of domain-dependent implementation effort in spite of claims of portability.

One important parameter of natural language systems is the time needed for interaction. Most systems operate rather slowly. Two extremes in philosophy for dealing with this problem are to write a fast system that has less intelligence or write a slow system that does language understanding "properly". The slow systems, however, can be very slow, taking up to several minutes to respond to an utterance. The problem here, besides being too slow to be practical, is that little is learned about the dialog process. The characteristics of dialog change with interaction time, so the less like "real time" (i.e., like human understanding time) the system is, the less realistic the dialogs will be. On the other hand, computer technology is changing so rapidly that it is a reasonable research strategy to write software whose hardware demands will converge with hardware capabilities at some point in the future. If this precludes testing with a system that responds in real time, at least the user is provided with a system that has more sophisticated capabilities.

One of the difficulties when we free the user from the need to understand the structure and contents of a very large database is that we also leave him without an understanding of which queries will be expensive and which inexpensive. If, for example, a user asks for the number of down time hours a plane had in a specific calendar month in the PLANES database, a simple retrieval of the value is done from the monthly summaries. If the same question is asked for a one month period starting on the 15th, the daily records must be accessed, making the search roughly 30 times larger. This search could be horrendous for queries over several years of data that must use job cards or daily summaries. Since the user is presumably freed from worrying about such details, it becomes important to inform him of the consequences of his request. The response could take from milliseconds to tens of minutes on our small subset of the database. On the real database that ours was abstracted from, one request may involve numerous tape mounts, and could take hours for one request. This is a direct result of having a natural language interface which could allow a user to inadvertently request information that may take a very long time to gather.

The interaction environment may have an effect on the language they use on the system (a poorly designed environment can induce frustrated obscenities from even the most patient users). The users of ROBOT, for example, rarely use quotas on their database requests. The reason is that all data that is

retrieved is displayed as it is found (not after all the data is found) and the display is paged. When the screen is full, the user has the option of aborting further retrieval, thus eliminating the need for phrases like, "at least ten values". ROBOT also has a time-out feature that asks the user if he wants to continue the search if it has already taken longer than about 10 seconds. A handy adjunct to this would be the capability for getting status information on the progress of a long search (one hates to abort after five minutes -- the end could be ten seconds away, although it may be 20 minutes off.) Finally, while most natural language question answerers operate on single sentence utterances, RENDEZVOUS has a terminal editor that allows the user to compose more extensive requests. After the request, often consisting of several sentences in their examples, has been composed and edited, a special SEND key is pressed, sending the request off to the language understander. This arrangement may have the effect of inducing more elaborate descriptions of requests from users.

The last implementation issue relates to human factors considerations for the interaction. Information should be clearly displayed, and inputs should be editable. The tasks that users will be carrying out on the system should be considered, and workspaces, temporary files and so on should be provided for if necessary. Some of the users in the completeness testing of PLANES expressed the desire to "peel

off" a query and let it run in background while shorter queries were run in foreground (it is interesting that this request came from users who had virtually no knowledge of or experience with computing).

Implementation Issues

- Implementational closure
  - who assumes the burden of closure?
  - over syntactic forms
  - over word recognition
  - over semantic/pragmatic analysis
    - representational consistency
    - conceptual inheritance
    - inference
  - rule centralization
  - general rules/specific rules/exceptions
  - over mathematical function clusters
- Non-determinism and uncertainty
- Ambiguity
  - lexical
    - part of speech (dog, duck, buffalo, cow, badger are all verbs!)
    - context dependent (check)
    - common/technical definitions (failure, charm)
    - general words (make, perform, take)
      - how many (retrieve, count, sum)
      - and
      - prepositions
  - syntactic
    - modifier attachment (I took the money from Bill)
    - noun-noun modification
    - reduced relative clauses
  - semantic/pragmatic (the police couldn't stop gambling)
- Accuracy of responses
  - misunderstanding
    - ambiguity
    - error
  - precision
  - recall
- Handling partial understanding
  - specific problem indication
  - unknown words
  - unknown constructions

Figure 4 Taxonomy of Implementation Issues

- Pragmatics of dialog
  - goals and plans
    - corrective indirect responses
    - suggestive indirect responses
    - supportive indirect responses
    - data discontinuities
  - believability
- Restricted subdialogs (clarification, tutorial, help)
  - menu selection
  - light pen, touch panel, track ball, special function keys
- Vocabulary size (too many words for string space)
- Data display
  - English generation
  - lists
  - tables
  - diagrams
  - graphs
- Portability
  - initialization and maintenance expertise
    - system generator
    - system editor
  - compatibility with existing systems (eg. databases)
  - domain dependence
- Interaction time
  - thinking
  - entry
  - processing
    - interpretation
    - retrieval
- Operating cost, and informing the user of
- Design of the environment
  - paged display (eliminates need for quotas)
  - timeout on long searches
  - xmit character (multiple sentence questions)
  - interrupt facility and status info
- Miscellaneous human factors considerations
  - information display strategies
  - user input editing facilities
  - workspace, temporary files, notes
  - control structure (peeling off long searches)

Figure 4 (cont.) Taxonomy of Implementation Issues

# Chapter 5

# AN EVALUATION OF PLANES

This chapter describes the application of the evaluation techniques outlined above to PLANES, a natural language question answering system designed and implemented at the University of Illinois. The chapter will consist of a brief introduction to PLANES, followed by a general description of the tests and ending with a detailed description of the performance of PLANES.

## 1.0 OVERVIEW OF PLANES

PLANES (Programmed LANguage-based Enquiry System) [Waltz, 1978] [Waltz and Goodman, 1977] is a natural language

interface to a large database. The database is the Navy's 3-M database, which holds the maintenance and flight records for all naval aircraft. It is on the order of a trillion bits, and grows by the equivalent of 100 magnetic tapes of data annually. The purpose of the PLANES project has been to explore the possibilities of building a natural language interface to the database.

The normal operation of PLANES entails three steps. First, the user's question is typed in and it is analyzed for its semantic content. Next, a formal query is constructed. The formal query is used to generate a paraphrase of the user's question, as PLANES has understood it. Last, if the user accepts the paraphrase, the query is performed on the database and the results are presented to the user.

## 1.1 Semantic Constituent Scan

PLANES makes very little explicit use of syntactic information. Its analysis of a user's sentence is a left to right scan for "semantic constituents". A semantic constituent is a phrase that can (usually) be mapped directly into some part of the formal query, such as a predicate or return field. The semantic constituents include such items as PLANETYPE, TIMEPERIOD, MALFUNCTIONCODE and so on. The analysis is implemented with an ATN. The top level is a one state network which calls various subnets to analyze the input

for semantic constituents (this view is somewhat simplified but is essentially correct).

1) How many A7's flew three or more catapult flights in May, 1971?

In the analysis of example 1, the first semantic constituent found is the question word "how many," found in the subnet QWORD. Next, "A7's" is identified as a class of attack aircraft in the PLANETYPE subnet. ACTION recognizes "flew." A subnet for FLIGHTS identifies "three or more catapult flights" and TIMEPERIOD picks up "in May, 1971." These constituents are now held in registers. The only structural information that is retained is the order that the constituents were found in the sentence. Since there is a PLANETYPE between the question word and verb in example 1, PLANES will return the aircraft serial number field. However, because this is a "how many" question, PLANES will count the number of elements in the return field and display the count to the user.

The one state ATN at the top level calls each subnet repeatedly during the analysis of the sentence. This is done without regard to whether the constituent that the subnet parses has been identified elsewhere in the sentence. What this is, then, is a top-down analysis technique that does not make use of expectations at the top level!

This unusual approach to analysis does have some justification. One of the goals of PLANES was to be very tolerant of user inputs that are not necessarily stated in standard English. The idea was that the sentence level structure (syntax) of inputs would tend not to be well formed, while the semantic constituent level would be fairly grammatical. PLANES expected to see rampant pragmatic ellipsis with the deletion of verbs, case markers and determiners. It was designed to be sufficiently robust to identify semantic constituents and build queries in spite of such adversity. It was presumed that the difficulty of designing a detailed grammar for English that would condone such a cavalier attitude toward standard syntax would not warrant the advantages. Instead, it was decided to ignore most of the constraints of sentence-level English syntax.

The inefficiency of pushing to many subnets that do not eventually accept a phrase of the input sentence can also be improved. A predicate could be attached to each push arc to test the next word of the input string. If there is a string that could be parsed by the subnet that starts with the next word, the push would be executed. If the subnet under consideration could not parse a string starting with the next word, the push would not be made. A manual count of arc transitions indicated that this one word lookahead would reduce the total number of arcs tested in the parse of a sentence by a factor of ten. A lookahead to the next open

class word, overlooking prepositions and determiners, would result in a reduction of arcs tested by a factor of 100. This would resemble a bottom-up analysis strategy at the sentence level, but a top-down strategy when parsing semantic constituents in the subnets.

## 1.2 Concept Case Frames

After the semantic constituents of the user's sentence had been collected, the collection was examined from the point of view of whether it constituted a complete query as it stood. If so, it was passed on to the query generation routines. If not, the missing constituents were filled in from the context. The context in this case was the set of semantic constituents used in the previous query labeled by semantic type. If a TIMEPERIOD was missing as in,

2) How many A7's had unscheduled maintenance?

the TIMEPERIOD constituent from the previous query would be used again.

The decision as to whether a user's question actually specified a complete query without borrowing constituents from the context was made on the basis of "concept case frames." Concept case frames were unordered sets of constituents of reasonable queries. Example 1 is repeated below with its

constituents labelled.

```
How many      A7's    flew    three or more catapult flights
QWORD         PLANETYPE FLY           FLIGHTS

in May, 1971?
  TIMEPERIOD
```

For this query to be judged complete, a concept case frame whose constituents include the constituent types "QWORD PLANETYPE FLY FLIGHTS TIMEPERIOD" had to be found. (An exact match need not always be found, as will be described in the next section.)

### 1.2.1 Substructure Ellipses And Pronouns

There were several ways in which a mismatch could have occurred. First, the constituent between the QWORD and verb might be missing. This is labeled an ellipsis and the constituent filling that function in the preceding sentence is used. Second, one of the constituents could be missing, having been replaced by a pronoun. In this case, the concept case frame that most closely matches the pattern of constituents for the current sentence is used. If there is a one constituent difference between the current sentence and a concept case frame, that constituent is taken as the referent of the pronoun.

3) What maintenances were performed on plane 3 in May 1971?
[QWORD MAINTTYPE MAINTACTION PLANETYPE TIMEPERIOD]

4) What maintenances were performed on it in August, 1973?
[QWORD MAINTTYPE MAINTACTION PRONOUN TIMEPERIOD]

The question in example 3 specifies a complete query without recourse to context. A concept case frame would match its string of semantic constituent types. The string of constituents for example 4 would nearly match the same concept case frame. The previous PLANETYPE, PLANE 3, would be substituted for the PRONOUN constituent.

### 1.2.2 Pragmatic Ellipsis

Another kind of incomplete match to concept case frames is with the occurrance of pragmatic ellipsis.

5) What maintenances were performed on plane 48?
[QWORD MAINTTYPE MAINTACTION PLANETYPE]

The TIMEPERIOD constituent has been ellipted from example 5. The constituent string in example 5 would nearly match the concept case frame that matched example 3, the TIMEPERIOD being the only discrepancy. This could be filled in from context (ie., the TIMEPERIOD constituent of the previous query).

### 1.2.3 Superstructure Ellipsis

The last case of an incomplete match to concept case frames was in the event of a superstructure ellipsis.

6) **Plane 50?**
**PLANETYPE**

Superstructure ellipsis like the one in example 6 would be recognized by the absence of a verb. Instances of superstructure ellipsis were handled by using the entire constituent string of the previous sentence (after it had undergone pronoun and ellipsis resolution) and replacing the constituents that have been mentioned in the current sentence.

If for some reason the pronoun and ellipsis routines were not successful, a clarification dialog would be initiated with the user to remedy the problem.

## 1.3 The Query Generator

After application of the concept case frames, the semantic constituents were passed on to the query generator. The query generator produced query predicates from the semantic constituents, each constituent being interpreted independently of the others. The fields to be returned from the query, as mentioned above, were taken from the noun phrases that fell (or had been ellipted) between the question

word and verb. PLANES maintained a list of the files that each data field occurred in. The file lists of the fields mentioned in the predicates and return fields were intersected to determine which files needed to be searched. This process also identified more involved queries, such as those involving joins.

After the query had been built by the query generator, it was given to the paraphraser. The paraphraser generated an English paraphrase for the user's approval. Upon approval, the query was executed and the results were returned to the user.

## 2.0 DESCRIPTION OF THE TESTS

Two sets of tests were con for the purposes of evaluation of PLANES. The first habitability testing, in which the users were given problems to solve using PLANES itself. The second was for completeness testing in which the users communicated with a person through terminal, who in turn generated database queries to answer their questions.

### 2.1 The Users

The users were students from the Aviation Institute at the University of Illinois. They were all due to receive certification for airframe and powerplant maintenance within a

few weeks. All were familiar with civilian flight and maintenance records. All were pilots, with an average of 218 flight hours. In short, they were familiar with the domain of discourse, aircraft flight and maintenance. Before the tests, each user was given a brief written description of the kind of data that he would have access to through PLANES (in the case of habitability testing), or through the human intermediary (in the case of completeness testing). This description was intended to introduce the users only to the kind of data available in the system. There was no information on the structure of the database, nor any information on how to use PLANES. The goal was to have users familiar with the domain, but completely naive about the natural language system.

## 2.2 The Problems

The users were given a set of problems to solve using the system. The habitability users were given a set of seven problems. Each user received the same set of seven problems, but they were given in random order. The habitability problems were selected to be fairly easily solvable. They involved simple retrieval and "how many" questions (which required counting, retrieval or summing -- the user was unaware which strategy was needed to answer a particular question.) An expert PLANES user could have solved the seven problems with eight questions. The users averaged 28 queries

per session (the sessions lasted from 1.5 to 2 hours). Four of the 14 habitability users answered all seven problems. The average number of problems solved was 5. The problems were stated in a way that was intended to minimize the effects of the language of the problem on the language of the users. The problems were stated as requests to fill in tables or make histograms. The users had to infer from the axes of the empty graphs or the labeled rows and columns of the empty tables what information was being requested.

The completeness test users were given one problem (each was given the same problem) which was much more open-ended. It asked the users to investigate the fact that the average NOR hours per plane for 1971 was 25 percent higher than in the other years. The open ended nature of this problem encouraged more variety in the kind of utterances that the users generated. A simple retrieval problem carries with it the implicit presupposition that the user can simply ask for the data and have it returned. With a problem such as the one given to completeness users, that presupposition is evidently not present, since no one simply passed the task on to the system (i.e., no one said, "Investigate the high NOR condition in 1971.") The users applied their knowledge of the domain and their problem solving strategies to conduct their investigations.

The problem given to the completeness users was much more like the kind of problem that would confront a decision maker. The problems that were given to the habitability users were more like the kind of sub-problems that a decision maker might generate for himself in an investigation. I did not give the completeness test problem to the habitability users because the results of preliminary testing indicated that this would cause a lot of false starts by the user before he was able to determine what kind of utterances were acceptable to PLANES and which were not.

I feel that all the problems, including the one for completeness and the ones for habitability, were "realistic" in some sense. A study was done of the kind of data requests that would be made by users of the 3M database [NAILSC, 1974]. The problems used in the habitability test were very similar to those listed in the study. The problem for the completeness test is one that was taken directly from an appendix of the NAILSC report. The appendix described how a user of the 3M database should conduct a high NOR investigation. I examined the data that we had, and found that there was a high NOR condition in 1971, and used that as the completeness problem. I did not, however, give the users any guidance in how to conduct a high NOR investigation. I relied upon their domain knowledge to help them generate hypotheses about what the cause could be, then use the database system (with a human intermediary) to test them.

# 3.0 PERFORMANCE SUMMARY

## 3.1 Habitability Testing

The habitability testing involved 14 users for a total of 429 utterances. Of these, 27 elicited system bugs (explained below) which precluded analysis of PLANES' language understanding ability. The results of the remaining 402 utterances are summarized in table 1. Classifying a response to a user utterance as appropriate or erroneous is not a simple judgement in natural language systems. Some of the user utterances were not stated in what I would consider a clear fashion. Nevertheless, PLANES did its best to interpret them, and frequently produced responses which were not appropriate to the utterance. I listed these as inappropriate responses of the language understander. I did not blame them on the users. I feel that there are two reasons for listing them as errors rather than attributing the blame to the users. First, it is difficult to establish a reliable metric for the understandability of an utterance. Second, a natural language processor will be exposed to anomalous utterances, and so should be able to respond to them appropriately. If the utterance is incomprehensible, an indication of this is a totally acceptable form of response.

I listed certain problems as bugs in the system. The purpose of the evaluation is to better understand the language understanding capabilities of the system. The problems that

were listed as bugs are those which had little to do with understanding. For example, the most common bug was that when opening files during database searches, some could not be closed (this was later found to be due to a bug in the TOPS-10 operating system.) It would be pointless to fault PLANES for this.

| | |
|---|---|
| Total No. of Utterances | 429 |
| Bugs | 27 |
| No. of Processed Utterances | 402 |
| Properly Processed | 68.4% |
| Improperly Processed | 29.1% |
| Could Not Interpret | 2.5% |

Table 1
Gross Success Rate

## 3.2 Success Fluctuations Among Users

The success rates varied among the 14 users from a low of 39.1% to a high of 89.4%, and were fairly evenly distributed throughout this range (see figure 5). Some of the higher success rates could be attributable to repetitive utterances. One user asked, for example, "How many NOR hours did plane 4 have in Jan of 1973," then proceeded to ask the same question again for each of the twelve months. This yielded 12 appropriate responses. Another user asked, "List the NOR hours in each month of 1973 for plane 4," which yielded one

appropriate response. Others suffered a number of inappropriate responses by trying to avoid the repetition of asking the same question for each of twelve months. Utterances such as those listed below served to lower the success rates for their dialogs.

7) How many NOR hours did plane 4 have in the consecutive months of 1973?
8) How many NOR hours did plane 4 have in April May June July August September October November and December separately
9) How many NOR hours did plane 4 have for each month in 1973?
10) How many hours for each month did plane 4 have in 1973?

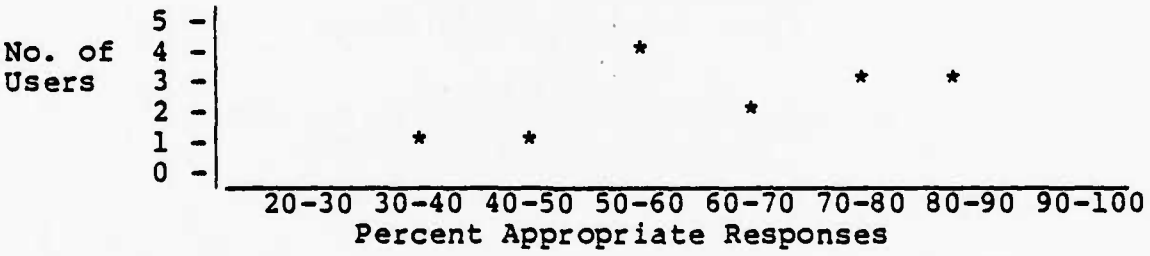



Figure 5
Distribution of Success Rates Among Users

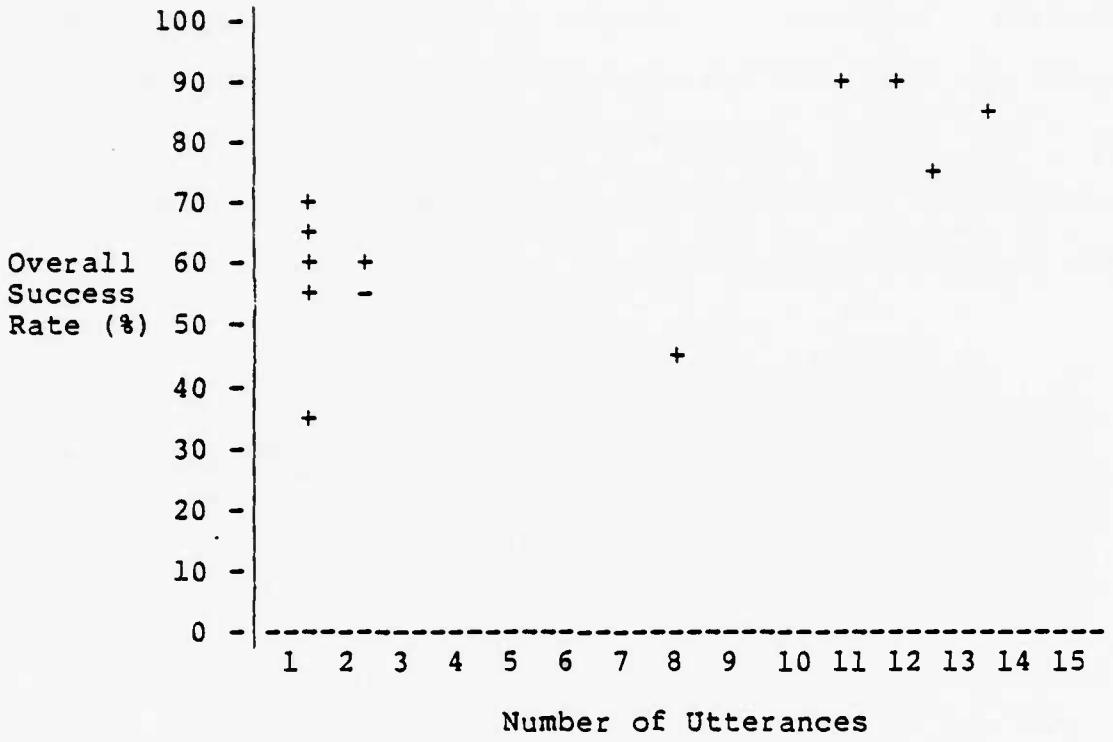



Figure 6
No. of Utterances to Find the No. of NOR Hours for Plane 4 for 1973

The question of success rate being in part a function of the repetitiousness of user's queries is explored somewhat further in figure 6. The database problem mentioned above has been used as an indicator of repitition. The number of queries needed to get the answer to this problem is plotted against the overall success rate for the users who attempted the problem. All but one of those who attempted the problem were able to get the correct answer to it. We find the highest overall success rates clustered around twelve

utterances to solve the problem. This corresponds to those users who asked for the NOR hours for each month individually. One user tenaciously tried to get the answer with a single query, but tried eight times before being successful. The users clustered in the one to two query region were the ones who were able to solve the problem with a single question, the more desirable method. However, their overall success rates are about 20% lower than the other group.

Another indicator of the inadequacy of the success rate measure is illustrated in figure 7. In this figure, the scores that users received on the problems they were trying to solve is plotted against PLANES' success rate in understanding their utterances. The problems were each worth ten points. One of the problems was not solved by anyone. Partial credit was given for an answer in which some of the data was present and correct. Full credit was given for fully correct answers.

One would hope to find a correlation between the utterance success rate and the score on the problems. This would indicate that the users who are being understood most frequently are making the most progress toward solving their database problems. This is evidently not the case. Many of the users who were understood most frequently made relatively little progress in solving the problems. It is interesting to note further that of the points that the users did not earn on their problem solutions, less than 10% were due to wrong

answers. The great majority were caused by the users running out of time, or trying to get data and failing (and knowing that they had failed).

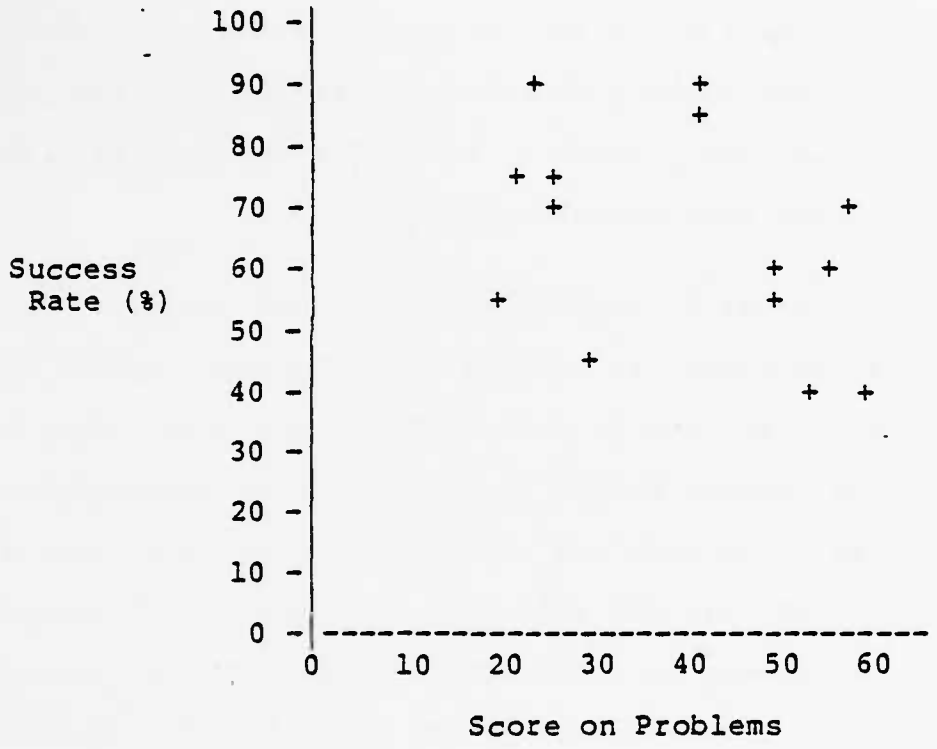



Figure 7
Score on Problems vs Success Rate

The important point of this is that it illustrates how poorly global statistics such as success rates describe language understanding capability. Success rate studies wash away the characteristics of the questions that were being asked, and treat each question as equivalent. Each question is not equivalent. One of the primary advantages of natural language is its ability to allow elaborate concepts to be

communicated succinctly. A natural language processor should support this ability, and a success rate analysis does not indicate it.

The results also illustrate another point, that of the limitation of habitability studies. If a user were communicating with a question answerer which he considered intelligent, it is doubtful that he would ask for data on a month-by-month basis. The dialogs collected from habitability studies are useful for determining how well the users can use the system as it stands, but they are not useful in determining whether the system provides what the users would like it to provide. They adapt too quickly to the limitations of the system. Completeness testing, on the other hand, is much more likely to reveal how well a system provides what the users would like to find in it. Another illustration of this fact is that the users who entered the repetitive queries retyped the entire sentence for each month. PLANES would have been able to understand if they had typed in the first query, then entered the months elliptically, as shown below, but none did. Surely, their language behavior was changed to fit what they perceived as the limitations of the system.

11a) How many NOR hours did plane 4 have in Jan, 1972
b) Feb
c) Mar

## 3.3 New Error Rate

New errors are those misunderstood utterances introduced by each user that are misunderstood for a reason different from those that preceded it. For example, if several users mentioned "hours of NOR", and the system did not recognize this phrase, it would only be counted as a new error the first time it was used. New errors are of interest because they indicate how well a system can satisfy the needs of a large user group. We would like to find that the number of new errors declines rapidly with each user. Figure 8 shows new errors plotted against users. New errors are counted only the first time they occur, even though the user who first introduced one may have caused it to occur many times in his session.

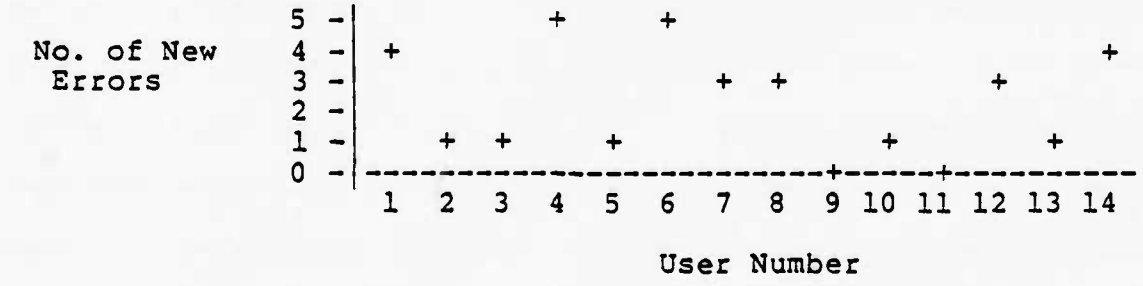



Figure 8
Number of New Errors Introduced by Successive Users

The data is too widely scattered to make any sort of reasonable statement as to whether it would settle down quickly. Data from more users would have to be collected in

order to determine this. I would suspect, based on my experience in observing user behavior with natural language processors that there will be a some quirks that new users will introduce even after the quirks of many preceding users have been made acceptable to the system.

## 3.4 Timing

The tests of PLANES were carried out on a DEC-10 with a KI10 processor. The load on the system varied from one or two other users to about twelve. The mean real time for a response was about 68 seconds, with a standard deviation of 39 seconds. Approximately 10% of this time was due to garbage collection. Parsing user utterances (the process from reading it in through preparing a formal query) took and average of 8.21 CPU seconds (standard deviation, 3.29 seconds). The database retrieval took an average of 1.84 seconds (standard deviation, 2.28 seconds).

| | CPU Time | Real Time |
|---|---|---|
| Parsing | 8.21 sec | 56 sec |
| Retrieval | 1.84 sec | 12 sec |
| Total | 10.05 sec | 68 sec |

Table 2
Timing

## 4.0 ASSUMPTIONS ABOUT THE USERS

No assumptions were stated explicitly about the users of PLANES prior to this work on evaluation. Some assumptions were indicated indirectly by the design goals of the system. The users were expected to be familiar with the domain of discourse. They were expected to be familiar with the esoteric vocabulary of military maintenance, such as NOR, RMCNFE, and FPC. References to such concepts were identified by their names, but could not usually be identified by any other description. There was also an implicit assumption that the users were fairly familiar with the database. The evidence for this is that users' utterances were assumed by PLANES to be meaningful as database queries. If they did not parse properly, the problem was assumed to be with the PLANES parser, and not with the users' utterances.

The restricted domain was used extensively to simplify the language understanding task. Some general words (eg., "make" and "occur") were given very domain specific interpretations. This indicates that the users were expected to stay within the confines of a restricted domain of discourse, and not use these general words except with their domain specific senses.

The users were assumed to be unfamiliar with the capabilities of the language analysis component, but to expect that it could understand whatever they entered. The users

were expected to use fairly succinct and telegraphic language. PLANES was prepared to interpret various forms of elliptical utterances.

The users were assumed to be ignorant of the organization of the database. PLANES did not accept references to data files, relation names, data domains or other concepts specific to the structure of the database.

The users were assumed to be unfamiliar with PLANES when they started using it. It was hoped that knowing that this was a natural language interface to the 3M database, and having a general understanding of the domain would enable the users to use the system without prior instruction. There was an on-line help facility that would be available to users who had difficulties in using the system (but the help system was not available to test users).

## 5.0 CONCEPTUAL COVERAGE

The topics discussed in this section will (more or less) follow the concepts listed in the taxonomy of conceptual coverage given in chapter 4.

## 5.1 Concepts From Closed Class Words

The semantic constituents that the parser discovers in a user's utterance are mapped to domains or data values in the relational database. PLANES does not have a representation for the concepts indicated by the closed class words, except for time. PLANES recognizes a wide variety of time phrases, and can represent either points in time or time spans. It cannot, however, represent a list of time points or spans. In the preliminary testing of PLANES in preparation for this study, a problem was given to users asking for data from two consecutive days. If the question was phrased in such a way that the two days indicated a span of time ("between May 16 and 17" or "from May 16 to 17"), PLANES could interpret it. The users instead tended to indicate this as a list of points or spans of time as "on May 16 and 17". Without intervention, these users would attempt many variations on their utterances before attempting and understanding that the time expression should be rephrased into one of the more awkward forms given above. In the selection of problems for the user tests, references to time that might involve lists of points or spans was avoided simply because the users would spend too much of their sessions trying to understand this one limitation.

## 5.2 Domain Specific Concepts

The limits of discourse for PLANES were roughly defined to be the contents of the database. This is in agreement with most other question answering systems. PLANES did have one facility, however, that enabled questions beyond the limits of the database. A semantic grammar parser was attached to the front end of PLANES which was designed to recognize utterances other than database queries. For example, a user could ask for definitions of terms or give some commands to PLANES such as "turn the paraphraser off." There was also a facility for defining new terms as synonyms of existing terms or phrases.

The preprocessor worked as a literal phrase recognizer. As such, it captured few if any generalities. Particular phrases could be recognized using it, but a similar phrase, say a syntactic variant or a slight rewording, would not be recognized unless it too had been explicitly added as a recognized phrase. When an entire phrase had been matched, a procedure at the end of the phrase in the ATN would be invoked for a response. One typical procedure printed word definitions.

It also had one minor bug -- it did not check for the end of the utterance when a phrase was recognized. As a result, when the initial substring of an utterance matched one of the phrases in the preprocessor, the remainder of the utterance would be ignored and PLANES would generate the response

appropriate for the matched string. This caused problems on the following utterances.

12) What is the time taken for flight from June 26 to June 30 1973
13) What type of malfunctions occurred on aircraft no. 4 from June 1 to June 7, 1973

PLANES responded to the first with the time of day, and to the second with a list of all the malfunction codes (numbers) known to the system and their nomenclature (eg., "burned out light bulb").

### 5.2.1 Knowledge About The Domain

PLANES is primarily capable of understanding references to elements of the database to which it interfaces. These elements represent database attributes and particular database values. Predefined sets of elements can also be referred to. For example, one may refer to the damage howmal codes - that set of codes which imply malfunctions that would probably be considered damage (a burned out light bulb is not a type of damage, but birdstrike damage is - of course, if the bulb burned out because of installing the wrong type of battery, it would become a damage, but this is beyond PLANES' capability). Also, virtual domains may be defined. The user may refer to hours of NOR as if it were an attribute. It is not, at least not directly. NOR is the sum of the following attributes:

not operationally ready due to supply, not operationally ready due to scheduled maintenance and not operationally ready due to unscheduled maintenance.

Through the use of the preprocessor, a smattering of other concepts can be referred to, such as definitions and lists of codes. These cannot, however, be referenced in the variety of ways that, say, a plane can. One could not refer to a definition with a pronoun, for example. The allowable forms of reference are inconsistent.

One can identify a number of objects in the question answering domain to which PLANES provides no means of reference. First are items in the query language and the database model. The user has no way of describing a quota for a query, as in sentence 14.

14) Which ten parts were removed most frequently.

The user has no facility for referring to particular relations in the database, although there may be an advantage in doing so. The database includes both daily records and monthly summaries. The information is redundant and can be inconsistent. In the PLANES domain, the user is not empowered to explicitly specify specific relations for a query to range over. However, he is not protected from having to know about the data model. PLANES generates a paraphrase of user queries

from the formal query prior to evaluation of the query. An advantage of paraphrasing at this point is that there is little chance of interpretation errors being introduced after the user has approved the paraphrase. The disadvantage is that the paraphrase very strongly reflects the highly structured form of the formal query and includes a number of details that relate specifically to the data model. In other words, if the user is to understand the paraphrase, he must understand the operation and organization of the database system at a different level than he is permitted to describe in his queries. A simple test of this situation in other systems is to see whether system generated paraphrases can be fed back into the language processor and be interpreted properly. In PLANES, they cannot.

### 5.2.2 Knowledge About The System

Users sometimes refer to themselves or the system during the course of a conversation. Frequently, this is done in order to better understand what the capabilities of the system are. To accommodate this, a natural language processor ought to be aware of itself and the user - at least to a limited extent. Several examples of references to the capabilities were found in the completeness dialogs (none in the habitability dialogs). Examples are shown below.

15) Could I have a comparison between the RMC time in 71 and 72?
16) Could I possibly find out which system required the most maintenance, and if not what information could I get as a break down of each of the maintenance items?
17) Is there any way that I could get some form of summary for the JCN's for 1971?
18) Is it possible to find out how many JCN's are recorded for a/c #34

These examples, taken out of context, appear as if they might just be indirect speech acts, not asking about capabilities but indirectly asking for the data. The contexts that they come from have a number of requests left unsatisfied due either to inability to satisfy them or to advising the user that the request would take very long to answer. The users were proceeding cautiously, trying to understand the capabilities of the system before asking for data.

Malhotra found similar requests. In his study of a simulated "perfect" natural language question answerer, a number of references were made to the capabilities of the system. A selection of these are given in examples 19-21.

19) Can you calculate percentages?
20) Can you give me data on product mix from each plant?
21) Do you have a forecasting model for demand?

The value of having the system capable of recognizing the identity of the user would be high in a domain like the scheduling domain of PAL. When the user refers to himself or a group that he belongs to, the system can determine specific

referents. SHRDLU had an entity, FRIEND, in the knowledge base, but its use was apparently limited to being a referent for the first person pronouns. There was no information about FRIEND, except if the user chose to enter something as in sentence 22.

22) The blue pyramid is my favorite.

This differs from the scheduling domain where the system must be aware of the particular person it is talking to, what his scheduling constraints are, what research groups he is a member of and so on.

### 5.2.3 Exceeding The Limits Of Discourse

Most natural language processors are implemented with the assumption that the user will confine his discourse to the limits that the system was designed for. If the user steps outside these limits, he is simply told that he has not been understood. PLANES makes this assumption (except for the phrase recognizing preprocessor).

Both PLANES and the SOPHIE semantic grammar have the capability of skipping words that are not recognized. Note, however, that this does not allow the user to stray from the limits of discourse and be understood. Both systems assume that the user's utterance is intended to be confined to within

the limits of the domain, but that he has used some vocabulary that is not known (and not important) to the system.

RENDEZVOUS is the only system (that I am aware of) that deals directly with cases exceeding the limit of discourse. RENDEZVOUS has a list of keywords that, when seen, indicate that the user has exceeded the limits. No further processing is attempted on the utterance. The decision to abort processing is made solely on the presence of a keyword. For example, in the parts supply and shipments domain, the occurrence of the word "managers" triggers the response that there is no information on managers in the database.

Limiting the domain of discourse to the contents of the database, and assuming that the user will limit his utterances in a similar fashion adds a severe limitation on how "naive" a naive user can be. Clearly, these system designs do not assume that the user is naive about what data is in the database.

When a human is answering questions, it may happen that he understands nothing of what he was asked. Frequently, however, he would understand the asker's question, but not be able to answer it. Also, he may be able to understand nearly all of the question, and be able to enter into a clarification dialog to understand the rest. When the question is fully understood, he could then evaluate whether he can answer it. The advantage that the human question answerer has over the

systems mentioned above is that the asker can approach the human with a broad spectrum of ignorance and misconception, and be guided toward the services that the answerer can provide. A natural language processor that could do the same could extend its utility to a much broader class of users.

## 5.3 Hypothesis

PLANES did not support hypothesis, and the need for it did not arise in habitability testing. Those users were simply retrieving data, so there was no need for hypothesis. The completeness users, on the other hand, were involved in a higher level problem solving process in which they needed to generate hypotheses and test them against the data. For the most part, they did not express their hypotheses to the system, but it was evident from the dialogs that they had formed hypotheses. One user did take the time to explain why he wanted the data that he had requested.

There were no attempts to say something such as, "how would things have been at time X if we had done so-and-so at time Y?" The processes involved in aircraft maintenance are not understood well enough to be able to model its dynamics. In Malhotra's management domain, there are predictive models that can be used to check hypotheses. His data included several examples of hypothesis formation.

## 5.4 Logical Propositions

The predicates that were generated from each semantic constituent were assumed to be logically ANDed. However, within a semantic constituent, predicates could be logically ORed. For example, if data were requested for planes 1, 2, 3, 4, and 5, the predicates for the aircraft serial numbers would be ORed (i.e., data on any and all is retrieved). This is because they are all parsed within the same semantic constituent. If data were requested for "plane 1 on June 3 and plane 2", then the predicates for the two serial numbers would be ANDed, which would, necessarily, yield a null response (there is no plane whose serial number is simultaneously 1 and 2). This is an admittedly improbable construction.

PLANES had no facility for negation, and no need for it arose in the testing.

## 5.5 Quantitative Propositions

Numerical quantification was handled in PLANES by checks written into each appropriate semantic constituent parser for a quantification phrase. It would be more satisfactory to factor out the commonality, and allow a numerical quantifier to ap ear before any noun phrase, but PLANES was not implemented this way. In fact, I know of no formalism for

semantic interpretation in which such regularities are factored out into a general statement (such generalization is common in syntactic parsers, but not in semantic interpreters.)

Only one numerically quantified noun phrase occurred in the habitability dialogs. It is listed as example 23.

23) did plane 4 fly only three missions from june 26 to june 30, 1973

The quantification phrase was not interpreted correctly since quantification was not anticipated for the noun "missions". The "three" was assumed to be a HOWMAL code. The "only" was ignored.

The problems that were given to the habitability users did not require the use of numerical quantifiers. The problem given to the completeness users also did not require quantification, but did encourage it. There were frequent references to "the other three years" when comparing data for 1971 to 1970, 1972, and 1973. Also, one user, in reaction to having a long list of data scroll off the top of his screen before he could read it, put a quota on her request.

24) List top ten locations that had performed the most mat. over the four years.

The problems that the test users were given did not require the use of comparative constructions. PLANES does accept comparative constructions. Comparatives and numerical quantifiers were required for some of the problems given to users in the preliminary tests, and it was found that a major difficulty in understanding them was clearly the fault of the users. Users put numerical quantifiers in seemingly random positions in the sentences.

## 5.6 Numerical Functions

PLANES supported addition and counting in so far as it was required to answer how-many questions. It did not support any numerical functions which could be called explicitly. Neither did any "hooks" exist in the internal representation which would allow the easy inclusion of numerical functions. If one were only to consider parsing, the addition of some coverage of numerical functions would be fairly straightforward, merely adding some new constituent parsers. Many calls were made in the completeness dialogs for sums and averages of data.

## 5.7 Concept Association

Other than the phrase recognizer preprocessor, the domain of discourse was limited to the contents of the database.

Furthermore, within that domain, discourse was limited to the literal contents of the database. There was no facility for inference. There was no facility for identifying implicit relationships between data elements. This was not a serious problem in the PLANES domain because there do not appear to be any such relationships.

An example of the problem of pragmatic relationships between data elements was seen in the domain of the Automatic Advisor [Tennant, 1977]. The Automatic Advisor answered questions about the engineering courses at a university. When asked,

25) How many courses deal with computers

the Automatic Advisor would return one course. It was the only one that explicitly mentioned "computers" in its list of topics. However, about 15 or 20 additional courses dealt with specialized topics such as computer graphics, computer architecture, numerical analysis and computer vision. In order to answer this question correctly, the Automatic Advisor would have had to infer that computer graphics, numerical analysis and the others are specialized topics, all relevant to computers.

Once identified, the particular problem of representing the pragmatic relationships between data elements can be dealt with in a straightforward manner. These relationships could

be captured in a hierarchical structure of topics, identifying computer vision, for example, as a subtopic of computers.

As simple as this is, most natural language processors would have the same limitation in the courses domain. LUNAR, LIFER, PLANES, Reader, RENDEZVOUS, ROBOT, and SOPHIE would all have this limitation. The PA system and possibly SHRDLU would probably be capable of handling the problem immediately, but since it is not mentioned in the literature describing these systems, this is speculation.

The difficulty in adding a capability for handling pragmatic relationships to the systems mentioned above would depend upon the nature of the interface between the language processors and their knowledge bases. The inference itself would probably be best handled in the knowledge base, but the motivation for the inference probably should come from the semantic interpreter. Thus, the semantic interpreter would have to determine that "computers" in sentence 1 implies the general topic and all subtopics, while "computers" in sentence 2 just implies the physical objects themselves.

26) What courses deal with the design of computers.

PLANES would be incapable of differentiating these intended senses of "computers" since it must make a semantic interpretation of the semantic constituent in which

"computers" occurs, without the benefit of the larger context in which it appears.

## 5.8 Discourse Objects

Discourse objects comprise an important class of objects which PLANES (and many other natural language processors) is unable to reference. Discourse objects include objects and aggregates of objects described in previous discourse (in particular, answer sets from previous questions), topics of conversation, sentences (sentences themselves rather than the objects or events that they describe) and processes that have been carried out in the course of a dialog. These will be discussed in turn.

Throughout the course of a dialog, particular objects and aggregates of objects are constantly being identified. PLANES is more capable of identifying objects or aggregates through a complete description than it can through a repeated reference. For example, when asked,

> 27) Which A7s that had more than 20 flighthours in June had less than 10 NOR hours,

PLANES would be able to identify that set of A7s, and return the set to the user. However, the set could not be referred to again, as in sentence 28.

28) Did any of them have more than 20 flighthours in July, too?

The only objects that can be referred to by repeated references in PLANES are objects that are parsed as single semantic constituents in the preceding question. "Them" in sentence 27 refers to the restricted set of A7s described in the previous sentence (it happens to be identical to the answer set in this case). "Them" does not refer to a single semantic constituent, so PLANES could not resolve the reference.

Other natural language processors that produce a parse (syntactic or semantic) of the entire sentence have these objects available for later reference. In LUNAR, for example, every noun phrase that described a variable in the database query representation was retained for later reference.

29) What is the average concentration of aluminum in each breccia

In example 29, the noun phrases available to LUNAR for anaphoric reference are "the average concentration of aluminum in each breccia," "the average concentration of aluminum [in the overall phase]," "[the overall phase]," and "aluminum" (the phrases in brackets are generated by the semantic interpreter). The "breccia" noun phrase apparently cannot be referenced anaphorically, as would be intended by "them" in

sentence 30.

30) give me the plag analyses for them.

The theory of focus described by Grosz [1977] provides a broader mechanism for anaphoric reference. Objects available for reference are those described by a knowledge representation of the ongoing dialog, which is coupled with the system's knowledge representation for the world. As a dialog progresses, certain objects, those in focus, are selected to be the primary candidates for anaphoric references. In addition, a number of other objects, related to those in focus, are selected as members of a secondary focus. Mention of a house, for example, would bring the house into focus with parts of the house (living room, dining room) in secondary focus.

PLANES did not represent in any way the fact that the dialog that was being conducted consisted of particular elements. It had no representation for a topic of conversation, for instance. The effect was that the user could not refer to the topic or specify constraints on the topic for an upcoming dialog. This is closely tied to the query-per-utterance assumption. If each utterance is considered a query unto itself, except for pulling antecedents from the last question, there is no need for maintaining the topic of conversation. However, the query-per-utterance

assumption is not a particularly good one, as discussed elsewhere in this thesis.

PLANES is not unique in this respect. Most implemented natural language processors have not included this capability. Three exceptions are SHRDLU, Reader and PAL. It is interesting to note how the domains of discourse for these systems have encouraged the designers to include these capabilities. It is easy to think of question answerers as simply answering questions. However, the three systems described above are object builders or robotic systems. SHRDLU builds structures from blocks, Reader builds programs and PAL schedules appointments, sends messages and so forth. In these domains, the desired result of the dialog is often a complex item that cannot be easily expressed in one sentence. The user must "set the stage" or build the objects piece by piece. These domains require an awareness that a process is underway, and that successive sentences are steps in the process. The question answering domain also involves a process linking the sentences of a dialog, albeit a less obvious process.

In the preliminary testing, some users referred to the process of the preceding dialog itself. In other words, a user may have a goal of assembling a certain body of data, then processing it in a particular way, such as finding the flight and maintenance records for a plane over a year. He

may then wish to repeat the process for another plane. To do so, he must be able to refer to the process itself. PLANES does not support such references, and neither do any other contemporary natural language processors. This did not occur in either the habitability or completeness data.

One final discourse object of interest is sentences. Example 31 refers not to the meaning of a sentence, but to the sentence itself.

31) What was my last question

Once again, this occurred in the preliminary testing, but not in the habitability or completeness testing. PLANES cannot handle such a reference, and I do not know of any system that can. However, it is an important part of the shared user-machine environment, and perhaps should be included as a referable object in the system.

### 5.8.1 Goals And Discontinuities

PLANES answers wh-type questions literally as well as directly. The data is retrieved, formatted and displayed for the user. Siklossy [1977] has pointed out that such answers are frequently misleading. Sentence 32, from the preliminary test dialogs illustrates such a case.

32) a. How many flighthours did planes 1 and 2 have in June?
b. 27

PLANES interprets sentence 32 as asking for the total number of flighthours between the two planes. It responded 27 hours, but the fact was that plane 1 had 27 hours while plane 2 didn't have any. Siklossy would call this a discontinuity in the data. He claims that humans expect to be notified of discontinuities, and feel misled if they are not. PLANES does not notify the user of discontinuities, and neither do other current natural language systems.

## 5.9 Extensions

PLANES has the ability to accept statements which make one phrase equivalent to another. Before parsing is attempted by the semantic constituent parsers, a pass is made over the sentence, substituting these phrases for their equivalents. No parsing is done on the phrase or its substitute, so no transformations can be made for changes in tense, number or slight changes in wording (such as changing the order of prepositional phrases). This feature would probably be most useful in assigning abbreviations to words or multi-word names.

Perhaps the most sophisticated facility for extensions is in REL. In this system, the user defines a new term in a quasi-English form (definitely a format that must be learned -- it would not be spontaneous. The definition is parsed, and can contain variables and semantic types of allowable phrases. Hence, this allow paraphrases to be generated of the extension, rather than just a fixed phrase substitute.

## 6.0 LINGUISTIC COVERAGE

The most striking feature of PLANES is its apparent lack of any sort of explicit syntactic analysis. It is important to determine the implications of this, from the point of view of both understanding the linguistic coverage of PLANES and understanding the advantages that PLANES or other systems could gain through syntactic analysis.

The language recognized by PLANES is not totally devoid of syntactic constraints. Each of the semantic constituents is recognized as a short, well-structured phrase. The semantic constituents that are recognized are syntactically well-formed. This is so simply because no syntactically ill-formed phrases are represented in the constituent subnets.

The area in which syntax is ignored is at the level above that of the semantic constituents. It is the level at which a query is formed from the set of semantic constituents. The

semantic constituents are a set of several different kinds of phrases. Determiners (question words), noun phrases, partial noun phrases, auxiliary verbs, main verbs, adverbial and adjectival prepositional phrases can all function as semantic constituents. In general, the subject, verb and object are not distinguished as such, a main verb may occur in a constituent separate from its auxiliaries and it is always separated from its object, and prepositional phrases can be separated from the words they modify (adverbial pp's always are separated, adjectival pp's often are). A determiner functioning as a question word is separated from the noun phrase it belongs with, but a special check is made to identify the constituent immediately following the question word. If it is missing, ellipsis is indicated. If it is present, it is taken as the intended return field for the query.

There are many meaningful and useful queries within the PLANES domain in which the absence of syntax in PLANES does not degrade the system performance. However, there are situations in which problems frequently arise. They are prepositional phrase attachment, word sense selection, non-commutative relationships and antecedent candidate nomination. This section will discuss the linguistic elements enumerated in chapter 4, but concentrate more heavily on those elements which caused the most serious difficulties.

## 6.1 Reference To Concepts

### 6.1.1 Reference By Name

If a semantic constituent can be identified by a simple name, PLANES has no difficulty in recognizing it. The constituent parser just accepts the name as a reference to the concept. This is where all natural language processors fare the best.

### 6.1.2 Reference By Description

If a description of a semantic constituent can be anticipated, and if it appears as a contiguous phrase in an utterance, a constituent parser can accept it as if it were simply a multi-word name. This is the mode in which PLANES is designed to operate. However, the descriptions are used in language to synthesize concepts. As such, not all descriptions can be anticipated by the system designers, so a natural language system that does not synthesize is destined to fail occasionally.

#### 6.1.2.1 Noun Modifiers

Predeterminers are accepted on a subnet-by-subnet basis. There are no general rules for handling them. Determiners are generally ignored. It was thought that person-number

disagreements especially involving determiners are so common in informal communication that it would not be beneficial to retain this information.

Ordinals and numerical quantifiers are parsed by a special subnet. It is called by each constituent parser that may take numerical quantifiers. This approach suffered from two problems. First, each phrase that can take numerical quantifiers must be specifically anticipated. No general rules were used to cover the most frequent cases. Second, it required that the quantifier and the quantified phrase be contiguous. Several examples of numerical quantifiers were found in the habitability dialogs that had not been anticipated in the PLANES subnets. Examples are shown below.

33) How many planes in each of the four years?
34) What was the average total flight time per aircraft in 1971 as compared to the other 3 years?
35) List the top ten locations that had performed the most mat. over the four years
36) I believe that you said that there were 36 different ACTWCs that performed service...
37) The parts supply condition probably played some part, but I don't think that is entirely responsible for the 25% increase of NOR
38) List JCN that have a NOR of greater than 8 hours
39) Can you determine if there was a high concentration of one type of HOWMAL and/or WUC for the 2 ACTORGS PE3 and AC2?

PLANES accepts quantification of NOR hours, but only in the prenominal position, thus the problem with example 38. Example 37 would not be accepted because it describes an increase, rather than a simple numerical value.

### 6.1.2.2 Logical Quantification

The design of PLANES largely ignored the issue of quantification. Several studies have addressed the issue, but no convincing guidelines for handling it have been forthcoming. A study by VanLehn [1978] even suggests that humans interpret sentences with complex quantification in an inconsistent manner, and are sometimes unwilling to choose an interpretation. This may suggest that humans are willing to accept quantification ambiguity until such time as they must resolve it or have it resolved by later references.

The main difficulties in the PLANES domain regarding quantification are when to return expanded vs. summarized data and where to perform a mapping function. The problems are related.

40) a. How many flights did each a7 have in May.
b. How many flights did the a7s have in May.

Sentence 40a indicates through the quantifier "each" that the monthly values for flights for May are to be returned, one per A7. PLANES does not consider the quantifier, so it would interpret 40a incorrectly. A heuristic in the query generation portion of PLANES causes "how many" questions to be answered by count or sum over an appropriate attribute. If 40a had been,

40c) What is the number of flights for each a7 in May,

PLANES would have returned the number of flights attribute, one value per A7. This would give the appearance of proper interpretation of the quantifier, but, as before, it is actually ignoring the quantifier.

Quantification is ambiguous in 40b. The user could have intended either one value per A7 or one value total. PLANES would return one value total.

41) a. What was the number of flights for each A7 in 1970
b. How many flights did each A7 make in 1970.
c. How many flights did the A7s make in 1970.

The data to answer question 40c is stored explicitly in the database. the data to answer questions 41a-c is not stored explicitly, but must be calculated from monthly totals. If the user's intent was to get one value per A7, which is indicated in 41a and 41b, then a mapping function must be applied which sums monthly values for each A7 to get yearly sums for each of them. Again, the quantification for 41c is ambiguous, but will always be interpreted as one value: a total. PLANES cannot perform mapping functions in its current implementation. It would interpret 14a as asking for a list of values per A7 per month. The monthly values for one aircraft over the year would not be summed. Sentence 41b would return one value: the total. PLANES cannot return one

value per A7 for this query.

Some of the responsibility for the inability to perform the mapping function indicated in 14a and b belongs with the query language. Most formal query languages permit specification of such calculations in the formal query. A new implementation of the query language for for PLANES is currently being written by Gary Brooks. It would permit the specification of mapping functions as well as other improvements.

One of the interesting findings of the testing was the ways in which quantification was expressed. Several constructions were used in addition to the familiar "each", "some", "every", etc. Examples are shown below.

42) List NOR hours for plane 4 in 1973. State the NOR hours for each month.
43) List separately NOR, flighthours for planes 13, 14, 15, 20, 22, 25 for June 1972
44) How many nor hours did plane 4 have in the consecutive months of 1973
45) How many nor hours did plane 4 have in april may june july august september october november and december separately
46) What were sum total flighthours for planes 2, 6, 45 and 48 in december, 1970
47) What were the individual total flighthours for planes 1, 2, 6, 45, 48 in december 1970.
48) What was individual number of flight hours during june 1972 for aircraft 14, 15, 22, 20, 25
49) What was the action taken for the different codes
50) What were the average NORMU hours in 1969, 1970, 1972 respectively?
51) What was the total number of flight hours for all the planes in 1971 added together?
52) ok, how about the AWM time, total for each year?
53) Was the higher nor from one of the aircraft, or was it from both aircraft combined?

54) Find NOR 1973 months no. 4
55) What was the quantity of parts of a given part number that were removed

## 6.2 Relative Clauses

Relative clauses pose serious problems for a parser that ignores syntax. The beginning of a relative clause is marked by a relative pronoun (example 56), a present or past participle (examples 57 and 58) or a noun phrase (example 59). They nearly always occur in the postnominal position in the noun phrase.

56) How many NOR hours were logged for the planes that had > 30 flight hours
57) How many NOR hours were logged for the planes having > 30 flight hours
58) How many NOR hours were logged for the planes scheduled to fly to San Diego
59) What were the part numbers of the parts the supply depot sent us on Nov. 1

Relative clauses can appear in other positions and can modify entire clauses [Quirk et al., 1976], but these forms are less frequent, and will not be discussed further.

Relative clauses cause difficulties for PLANES because it is difficult to determine where the relative clause terminates and the main clause resumes. PLANES uses the heuristic that a relative clause is either terminated by the end of the sentence or by a verb. The relative clauses in examples 45-59

all terminate at the end of the sentence. In example 60, a relative clause is followed by a verb.

60) Were any of planes that had > 30 NOR hours flying more than 20 hours per week

There are two fundamental problems with this heuristic. First, the lexical class of a word may be ambiguous without knowledge of its function in a particular sentence. The word "flying" in example 60 functions as a verb but in exampl6124 it functions as a noun.

61) Were any of the planes that had > 30 flying hours grounded.

Several other words that function as nouns and verbs in the PLANES domain of discourse are "crash", "damage", "malfunction", "repair", "schedule", and "work".

The second problem with this heuristic is that relative clauses may be followed by other constituents.

62) Were any of the planes that had > 30 flight hours NOR for < 5 hours
63) Print the unscheduled maintenance records for the planes that had > 30 flight hours in January on the line printer
64) Give me the daily NOR hours for the planes that had > 30 hours and sort them by serial number
65) Give me the planes that had > 30 flight hours last year
66) What are the A7s that had > 30 flight hours, the F4s with > 10 flight hours and the F11s having > 20 flight hours

Examples 62-66 show relative clauses that are followed by words that function as clauses other than verbs.

Once a relative clause has been bounded in PLANES, the head noun that it modifies is retrieved ("planes" in the preceding examples) and it replaces the relative pronoun if a relative pronoun is present. The clause is then parsed by PLANES as if it were an independent sentence. Semantic constituents are identified, analyzed with concept case frames, pronoun and ellipsis resolution is performed and a query is generated and executed. The relation returned by this query will later be joined to the relation generated for the rest of the sentence without the relative clause.

A new problem arises executing a query for a relative clause separately from the main clause. There is often information shared between the two clauses that may only be expressed in one of them.

67) Give me the NOR hours in May for the planes that had > 30 flight hours
68) Give me the NOR hours for the planes that had > 30 flight hours in May

The date phrase, "in May", is found in the main clause in example 67 and the relative clause in 68. However, these two clause are processed separately and so they do not share the date phrase. In each case, the clause without the date phrase would take it from the previous sentence. This may or may not

be appropriate, depending on the context. If the sentence before sentence 67 in a dialog were something like example 69, the interpretation would be heavily biased toward using April with the flight hour clause.

69) I am interested in the flight hours of all planes that flew in April

If sentence 30 was set in a more neutral context, May should probably be shared by both clauses. Example 70 illustrates a more neutral context.

70) How many planes are in the database.

Some questions pack nearly all the query information into a relative clause. Example 71 illustrates such a question.

71) What are the planes that had > 30 flight hours in May

Clearly, there is no need to attempt a query for the main clause of example 71.

## 6.3 Prepositional Phrase Attachment

The semantic constituent subnetworks recognize phrases on an essentially verbatim basis. In parsing "NOR hours" from example 72, the phrase itself was stored explicitly in an ATN

constituent net.

72) Did plane 3 have more than 50 NOR hours in May?

The recognized phrase is represented by a token representing the concept of NOR hours, and in this case, with the numerical quantification "more than 50". The quantifier phrase is parsed with a separate subnet that is called by the net that parsed "NOR hours". "Hours of NOR" can be parsed by the subnet as a phrase, just as "NOR hours" was, and once the concept is tokenized, the phrases "hours of NOR" and "NOR hours" are indistinguishable on the semantic level, as they should be.

73) Did plane 3 have more than 50 hours of NOR in May?

In the case of example 73, the prepositional phrase was appropriately attached to the word it modifies by virtue of the head noun and the qualifier being recognized as one contiguous phrase.

The seed for problems is planted in the analysis of examples 74 and 75 and grows in the subsequent examples.

74) How many hours did plane 3 fly in August?
75) How many hours was plane 3 NOR in August?

The word "hours" alone constitutes a semantic constituent in examples 74 and 75, and must eventually be interpreted (in this small domain) as the time, measured in hours, that a plane was in a particular state. The hours could be flight hours, or hours of NOR in general, hours of NOR due to scheduled or unscheduled maintenance, hours of NOR while awaiting the arrival of an ordered part, hours of NOR while awaiting maintenance or hours of reduced mission capability in general or for several particular reasons. The difficulty is that the interpretation of the phrase "hours" in examples 74 and 75 is determined by the interpretation of other parts of the sentences, parts which in this case, fall into semantic constituents other than the one containing "hours". But there is no mechanism in PLANES for determining which constituents designate the appropriate interpretation of "hours". This is the fundamental problem incurred when syntactic structure is ignored; when the interpretation of one constituent depends on the interpretation of others, syntax helps select the candidates that determine the appropriate interpretation.

PLANES has no mechanism for determining the interpretation of one constituent from that of others. It is forced to give a semantic interpretation as the constituent is identified. In the current implementation of PLANES, "hours" is interpreted as "flight hours". This interpretation is acceptable for example 74, but unacceptable for example 75. PLANES cannot properly analyze example 75.

The same situation arises when head nouns and prepositional phrases that modify them are separated. "Hours of NOR" is recognized as a phrase, but "hours for plane 3 of NOR" is not. The interpretation of "hours" depends on "of NOR". The "hours for plane 3 of NOR" is interpreted as three constituents that are not subsequently analyzed as a unified group. It would be simple enough to add the phrase "hours for PLANE-TYPE of STATUS" to a constituent subnet. A cleaner representation might be to allow the prepositional phrases to be parsed in any order by calling constituent parsers from the head noun node, then returning to the head noun node. The problem with this approach is that it produces two semantic constituents from one phrase, namely NOR-HOURS and BUSER-NUMBER = 3. There is no mechanism for this in PLANES, but it would be straightforward to add one.

Even with the prepositional phrase parsing improvement mentioned above, the problem has not been solved. Consider example 76.

76) How many hours did plane 5 have of NOR in August?

The noun "hours" is displaced from its qualifying prepositional phrase "of NOR" in this question construction. The scheme described in th above paragraphs for prepositional phrase attachment would not be of any service in example 76. To properly parse this question construction, the parsing of

the noun phrase headed by "hours" must be interrupted temporarily, then resumed at a later point in the sentence. In this case, the noun phrase parse must be resumed following the main verb, at the location of the object in an active voice construction (i.e., Plane 5 did have how many hours of NOR in August). The identification of the appropriate points in the sentence to suspend parsing and to later resume it is a purely syntactic decision. Parsing example 76 could not be accomplished in PLANES with any moderate alterations.

A final factor relevant to prepositional phrase attachment appears to work to advantage in PLANES. It is the frequently cited problem of prepositional phrase attachment ambiguity. In a phrase like example 77, it is syntactically ambiguous as to which structure the prepositional phrase modifies.

77) hours for the plane of NOR

"For the plane" could modify "hours" or another structure elsewhere in the sentence in which phrase 76 is found. "Of NOR" could modify "plane" or "hours" or another structure elsewhere in the sentence. In general, ambiguity in prepositional phrase attachment (or qualifier attachment in general) is a problem incurred with purely syntactic parsers. Attachment decisions can generally only be made through the consideration of semantic constraints.

PLANES has two advantages with respect to attachment ambiguity. The first is that the ATN subnets for parsing semantic constituents operate as phrase level semantic grammars. They embody both the syntactic and semantic constraints on phrases. Head nouns and their prepositional phrases are accepted by matching the phrase to ATN arcs which represent acceptable head-modifier combinations. The syntactic and semantic constraints are both satisfied because the phrase is matched word by word.

The second advantage that PLANES has in attachment ambiguity is that when a semantic constituent consists solely of a prepositional phrase, PLANES doesn't bother with attachment. The point of building queries from unordered sets of semantic constituents is that, in many cases, attachment is unnecessary.

78) (Give me)(the NOR hours)(for plane 3)(during August 1972)

The semantic constituents are bracketed in example 7. The query for example 78 is shown in 79.

```
79) (FIND ALL
    ((V  M))
    ((SUM(V(NOR))))
    (AND(EQU(V BUSER) 3)
        (EQU(V ACTDATEMON) 8)
            (EQU(V ACTDATEYR) 2))
     NIL)
```

Prepositional phrase attachment is irrelevant when

constructing query 79 from question 78. This advantage leads to the ability that PLANES has to analyze questions such as example 80. These ungrammatical or highly elliptical sentences will be discussed further in a later section.

80) NOR hours plane 3 August 72

## 6.4 Anaphora

Most discourse is quite coherent. The topic of each sentence, for the most part, is closely related to the topic of the previous sentence. Human speakers make use of this fact by abbreviating their utterances to a point where the omitted material can be readily recovered from the previous discourse. A sentence may be hopelessly ambiguous when considered out of context, but clear and understandable when embedded in an appropriate context. A presumption that refers to a concept or statement that has come before is anaphoric. Anaphora in various forms is the most frequently used method of establishing and making use of cohesive discourse. Two other methods are cataphora, presumption of items that will come later, and exophora, presumption of items in the general environment. This section will concentrate on anaphora.

The section of this chapter on conceptual coverage discussed anaphora, but from a different perspective. The point of that discussion was to identify the kinds of concepts that could be available for anaphoric reference. This discussion under linguistic coverage, will be concerned with the language in which anaphora is expressed.

Anaphora occur in two forms, those that reiterate previous text and those that refer to or relate to specific previously mentioned objects. Reference anaphora include the use of pronouns and definite noun phrases as in examples 81 and 82.

Lefty arrived looking famished
81) He suggested that we all go out to the Boar's Breath for dinner.
82) The jerk hadn't bothered to feed himself all day.

When the reader comes across "he" and "the jerk" in these examples, he knows that they refer to a particular previously mentioned person. In this case, they refer to the person named Lefty. Not just anyone named Lefty, but that particular person mentioned in the previous sentence.

Structural anaphora indicate the intended (understood, but not stated) reiteration of a fragment of the previous text. The anaphora may be indicated by a missing structure (ellipsis) or one that has been replaced with a word such as "one" (substitution). Example 83 illustrate ellipsis and 84

substitution.

Which planes flew more than 12 hours
83) Which flew more than 20 hours
84) Which ones flew more than 20 hours

In both of these examples a structural component of the original sentence was assumed. In both cases the speaker assumed that the hearer would understand the sentences as, "Which planes flew more than 20 hours." Thus a substructure was ellipted or substituted for. It may also be the case that most of a sentence is assumed, but not stated, and only a small structure is repeated. I call this superstructure ellipsis, and it is illustrated in example 85.

Which planes flew more than 12 hours?
85) 20 hours?

In this example, most of the structure of the intended sentence has been left unstated.

Anaphora are useful in language not only in making it more telegraphic but also in clarifying the speaker's intent. The use of reference anaphora is the only means at a speaker's disposal for referring to a specific previously mentioned concept. It is the only way he can avoid being misinterpreted as mentioning a new concept with a similar description (as with two individuals named Lefty in example 81 and 82.) Structural anaphora are very useful for focusing the

listener's attention on what is new in an utterance, or how it contrasts with what was said before.

One other phenomenon has been called ellipsis in the natural language processing literature. It is the case in which a sentence that may be acceptable syntactically is missing some information. In other words, the sentence would not make sense in isolation. I call this pragmatic ellipsis. It is illustrated in example 85.

How many planes flew more than 10 flights in June 1971
85) How many planes flew more than 20 flights

In the domain of aircraft histories, example 85 is not sensible, or at least highly improbable, without specifying a time period of interest. The time period is assumed to be understood without having been mentioned.

Pragmatic ellipsis is different from structural ellipsis in that it must be detected by a different process. Structural ellipsis and substitution is detectable syntactically or semantically. It is detected when the listener is trying to determine what has been said. Pragmatic ellipsis, on the other hand, is detectable only after the listener has determined the meaning of what has been said, and is in the process of determining whether it makes any sense. The request implicit in sentence 58 does not make sense unless a time period is specified.

The problems that anaphora pose for natural language processors are these: when has an anaphor been used, and what text fragment, structure or concept is the speaker expecting the listener to understand.

## 6.5 Anaphora In PLANES

PLANES supports four kinds of anaphora to some degree. It can handle pronominal references, substructure and superstructure ellipsis and pragmatic ellipsis. The degree to which each is handled will be discussed below.

The mechanism that supports all forms of anaphora in PLANES is the concept case frame. After the set of semantic constituents for a sentence has been identified, it is compared to the constituent in the concept case frames.

In general, several concept case frames will match the constituent set from the sentence. From these concept case frames, the one which matches best is selected as the appropriate one for the sentence. The best match is selected as follows. The semantic constituents from the sentence are paired with their counterparts in a concept case frame. After the known constituents have been paired, the remaining unfilled slots from the concept case frame are examined. If, for example, a MAINTENANCE-TYPE slot was left unfilled, the previous question is examined for a MAINTENANCE-TYPE slot

filler. The best concept case frame is the one that can fill the most slots from either the constituents of the current sentence or the preceding sentence. The slots in the concept case frames that must be filled by referring to the previous question represent phrases that have either been ellipted or pronominalized in the current sentence.

PLANES' approach to anaphora makes no distinction between referential anaphora (pronouns) and non-referential anaphora (ellipsis and substitution). All anaphora are treated non-referentially. PLANES also makes no attempt to treat definite noun phrases differently from indefinite ones (in fact determiners are, for the most part, ignored in the current implementation). Finally, no reference can be made to items in the answer to a previous question. The concept case frame approach only supports anaphora relating to the previous question.

One implication of not differentiating between referential and non-referential anaphora is that a description of an object cannot be accumulated over several sentences. Consider example 86.

> 86) I am trying to find out about an A7. It had 20 NOR hours in June. It averaged 30 flight hours monthly in 1971. How many flight hours did it average in 1972?

In the three sentences of example 89, the same aircraft is referred to repeatedly, and it must satisfy the conditions in

the first two sentences simultaneously. If "it" were taken as a phrase substitute, the description in each sentence would not have any particular relationship to one another, as shown in example 87.

> 87) I am trying to find out about an A7. An A7 had 20 NOR hours in June. An A7 averaged 30 flight hours monthly in 1971. How many flight hours did an A7 average in 1972?

It is difficult for us to read the sentences in example 87 without thinking something like, "oh, 'the A7' was intended," thus making it referential. The sequence suggests the description of one plane followed by a question about it. If PLANES could analyze these sentences (it can't handle anything but sentences that translate to database queries), it would interpret "it" non-referentially.

Example 86 illustrates another problem with the concept case frame approach to anaphora. The antecedent to an anaphor must always be one semantic constituent. Since nouns and their prepositional modifiers, or subjects and their verb phrases are generally parts of separate semantic constituents, the anaphor cannot be a substitute for larger structures composed of several semantic constituents.

This concept case frame approach is effective for referring to implied sets of concepts (even if the "reference" is actually done non-referentially!). For example, in example

88, "they" refers to the set of A7's, not the particular one described in the previous sentence.

88) Did the A7 with birdstrike damage have any flight hours in July?
Did they have more than 20 NOR hours?

PLANES would take "A7" as the antecedent. Unfortunately, if the pronoun were "it" instead of "they", it would make the same interpretation.

### 6.5.1 Query Synthesis

For the most part, PLANES expects the user to type in a question that PLANES will translate into a database query. The query will then be evaluated, and the data returned to the user. The user will then ask another question. The only exception to this cycle that has been provided for in PLANES is an utterance which matches an entry in the preprocessor.

In this way, PLANES is similar to other question answering systems. One exception is RENDEZVOUS. RENDEZVOUS permits the user to enter a description of his query that is as many sentences long as he wishes. The user indicates that he has finished his description by typing a special key. Furthermore, RENDEZVOUS paraphrases the user's utterance, then asks whether he would like to make any changes. This dialog is all highly structured. The same capabilities are inherent

in natural language communication, but it is affected with far greater subtlety. In human communication, ignoring changes in pitch and volume of the voice, the listener waits until the speaker has described a reasonable question. If there is something missing, the listener will wait for it. If it is not forthcoming he will ask for clarification. Changes by the asker are frequently made on an interrupt basis. So, in contrast to computer understanding, the less structured approach that humans take is heavily reliant on pragmatics rather than explicit surface cues like special keys and explicit questions.

PLANES answers wh-type questions directly. "Which planes ...," "What maintenances ...," "How many flights ..." are all answered as specific requests for retrieval, and nothing else. Yes-no questions, on the other hand, are always transformed into similar wh-type questions. For example,

89) Did any planes have more than 10 flighthours in June

sentence 89 is not answered "yes" or "no". Instead, a query is constructed which is equivalent to sentence 90.

90) What planes had more than 10 flighthours in June.

The motivation for this transformation is based on the assumption that in a question answering domain like this, the user is not interested in "yes" and "no" answers, but rather

in the data behind the answer.

The CO-OP system takes this idea a step further. When a query returns an empty response, CO-OP makes an effort to determine which predicate or combination of predicates in the query produced a null answer set. Thus, in response to sentence 90, a CO-OP-like system could respond that no planes flew in June.

## 6.6 Pragmatic Ellipsis

In an ongoing dialog, users will frequently omit phrases that are understood without mention within the context. Two forms of this are substructure and superstructure ellipsis, which are primarily syntactic-semantic phenomena. Some sentences that are syntactically complete and semantically interpretable may yet need further information before a meaningful database query can be built from it. I call this pragmatic ellipsis. It has also been included in the term world knowledge inference.

PLANES handles pragmatic ellipsis through the use of its concept case frames. Concept case frames represent questions that are pragmatically complete as well as syntactically complete. Missing semantic constituents are identified through the concept case frames. The constituents of the previous question are then searched for the missing

constituents, and, if found, are used in constructing a formal query. I know of no other question answering system that has attempted to deal directly with pragmatic ellipsis.

## 6.7 Speech Acts

As has been mentioned before, PLANES interprets almost every user utterance as a database query. The exceptions are certain sentences put into the preprocessor that give definitions etc. In the dialogs, a number of user utterances were not intended as database queries. However, nearly all of these occurred in the completeness test data. Two indirect speech acts occurred in the habitability data, but they were intended to be, and were interpreted as database queries. They are presented below.

91) I need to know the number of flight hours flown during June 1972 for aircraft with number 13.
92) Want number of flight hours flown by number 13 during June 1972.

These two sentences were intended to be interpreted as requests for information rather than simply descriptions of what the user wanted.

The completeness data shows many additional examples of sentences that were intended as requests for data, but stated in another form. It is interesting, however, that it also included a number of questions about the capabilities of the

system. The context determined which should be interpreted as which. Their structures were indistinguishable. These are discussed in the section on conceptual coverage.

In addition to indirect speech acts, there were speech acts other than requests for database searches.

93) What does NIL mean?
94) I don't understand these numbers.
95) Can I see that data again please?
96) I don't think I need that quite yet.
97) What I meant was, for example, did the a-7 experience more NOR time in 1971 vs. the other years, or was the increased NOR time due to both the a-7 and the f-4 being down more.
98) I believe that you said that there were 36 different ACTWCs that performed service on the aircraft so it is safe to assume that the work was the same performance wise...
99) Maybe there is a certain problem that kept happening.

Sentences 93 and 94 are questions about the information being displayed on the screen. Sentence 95 is a request to repeat the display process. Sentence 96 informs the system not to carry through a retrieval process. None of these requires or can be responded to with a database search. The intention behind sentence 97 was to clarify a previous remark, but it could be interpreted as a database request, complete unto itself. Sentences 98 and 99 are not requests to the database, but serve to clarify to the system what the user understands to be true.

It may not be the case that one would want to build a natural language system that would understand the kinds of utterances mentioned in this section. One could argue that they do not occur with sufficiently high frequency to warrant supporting them. However, they do illustrate a basic principle of communication -- that much of the language used in communicating is used so that the participants can better understand what the other knows, believes, and what he is confused about. Communication is not just, "Tell me X," where both know perfectly well what X is and what the answer means. To ignore this fact, is to attempt to model a characature of language that may occasionally be sufficient, but not always. If a one decides to build a system with such limitations, there should be some explanation made of what is expected of the user: is he to be told that such utterances will not be accepted, or will he be left to fend for himself and infer why he is not being understood (and at what cost of frustration)?

## 6.8 Negation

PLANES had no facility for handling negation. Neither was there any facility to recognize it when it occurred to flag it for the user. Phrases like "not operationally ready" were accepted as compound words, not as the negation of "operationally ready".

## 6.9 Conjunction

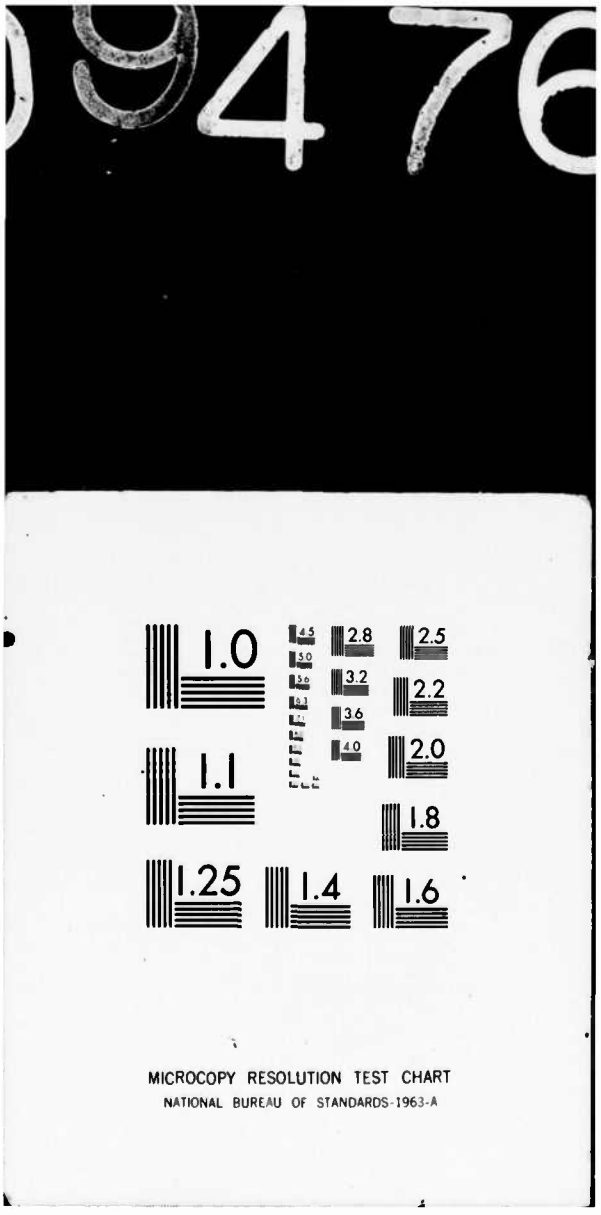
PLANES has no general conjunction capability, but does handle a number cases of it in the subnets. For example, the subnet that parses a PLANETYPE semantic constituent can accept a wide variety of conjoined forms of aircraft and aircraft types. Only rarely did a semantic constituent occur in which the conjunction was not understood. One mentioned previously was a reference to a list of dates, as opposed to a single date or a span of time between two dates. Another was the phrase "for plane one through 50," in which there is assumed knowledge that "one" and "50" designate the begin and end of an ordered set.

The semantic grammar approach taken in PLANES illustrates the semantic similarity of conjunction with other constructions. Dates, for example, can be accepted as conjoined as in "between June 1 and 7" or as a sequence of prepositional phrases as in "from June 1 to 7." It seems that this is a more fundamental regularity than the syntactic form of the phrases. That is to say that "from June 1 to 7" is more like "between June 1 and 7" than it is like "from Dallas to Chicago." More work is needed in semantic interpretation to integrate what is known about syntax with the ability to represent the similarity between the time phrases above.

AD-A094 762 ILLINOIS UNIV AT URBANA-CHAMPAIGN COORDINATED SCIENCE LAB F/G 5/2
EVALUATION OF NATURAL LANGUAGE PROCESSORS.(U)
NOV 80 H R TENNANT N00014-75-C-0612
UNCLASSIFIED T-103 NL

3 OF 3
AD
AD-A094762

END
DATE FILMED
3-81
DTIC


9476
1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

## 6.9 Conjunction

PLANES has no general conjunction capability, but does handle a number cases of it in the subnets. For example, the subnet that parses a PLANETYPE semantic constituent can accept a wide variety of conjoined forms of aircraft and aircraft types. Only rarely did a semantic constituent occur in which the conjunction was not understood. One mentioned previously was a reference to a list of dates, as opposed to a single date or a span of time between two dates. Another was the phrase "for plane one through 50," in which there is assumed knowledge that "one" and "50" designate the begin and end of an ordered set.

The semantic grammar approach taken in PLANES illustrates the semantic similarity of conjunction with other constructions. Dates, for example, can be accepted as conjoined as in "between June 1 and 7" or as a sequence of prepositional phrases as in "from June 1 to 7." It seems that this is a more fundamental regularity than the syntactic form of the phrases. That is to say that "from June 1 to 7" is more like "between June 1 and 7" than it is like "from Dallas to Chicago." More work is needed in semantic interpretation to integrate what is known about syntax with the ability to represent the similarity between the time phrases above.

There was no facility in PLANES to support conjunction when it did not occur within a semantic constituent. As a result, and utterances with conjoined clauses either failed completely or were misinterpreted.

## 6.10 Punctuation

Punctuation was ignored in PLANES. The only ill effect of this seems to be that when multiple sentences were entered, the entire string was parsed as one long sentence, and so was usually misinterpreted.

## 6.11 Operational Closure

### 6.11.1 Semantic Equivalence Classes

Most of the semantic equivalence classes listed in chapter 4 are based on syntactic transformations. PLANES ignores the order of semantic constituents in a sentence, so if the transformation can be interpreted in this way, it would be acceptable to PLANES. However, there is no explicit indication that a sentence structure X is semantically equivalent to another structure Y.

## 6.12 Discrimination

Since PLANES assumes that all utterances are database queries and that all semantic constituents can be mapped to elements of the queries, there is little cause for fine discrimination between concepts. Synonyms are taken as identical, for example. The issues of discrimination are intended to be relevant to a system that attempts a more thorough representation of meaning than what PLANES does.

# 7.0 IMPLEMENTATION ISSUES

## 7.1 Implementational Closure

There is little closure embodied in the formalism for PLANES. The closure that users observe is due to the diligence of those who implemented PLANES for its domain. There is little in the formalism in terms of written code that could be carried over to another implementation in a different domain, beyond the ATN parser. This seems to be a characteristic of semantic grammar based systems. What could be carried over is a collection of semantic constituent parsers that might find applicability in many domains. In particular, the date parser would be widely useful as would be the numerical quantifier parser. These represent a collection of a variety of ways to express these concepts. Another carry over would be the concept case frame identifier program. A

third is the spelling correction package.

## 7.2 Nondeterminism

PLANES is implemented in an ATN formalism, so it handles nondeterminism through the ATN backtracking facilities. Nondeterminism tends to be minimized in a semantic grammar system since each decision that is made is so specific. For example, it is much more specific to determine that the current phrase being parsed is a PLANETYPE rather than just a noun phrase. This information can be used in semantic grammar systems to restrict the choices later in the parsing (however, it simultaneously implies that there are more choices possible: instead of just a noun phrase, the phrase could be a PLANETYPE, MALFUNCTIONTYPE, DAMAGECODE, DATE, etc.) PLANES is particularly susceptible to problems of nondeterminism since it uses a top-down formalism (ATN), but does not use any sentence level expectations to guide later parsing. Each constituent parser works independently. As mentioned previously, some lookahead prior to pushing to a constituent parser could reduce the number of arcs traversed by a factor of ten to one hundred depending on the type of lookahead used.

## 7.3 Ambiguity

The current implementation of PLANES relies heavily on its restricted domain of discourse to eliminate most lexical ambiguity. Particularly severe cases are interpreting "make" as meaning "fly" and "perform" as "perform a maintenance action". Ambiguity of larger structures is assumed to be resolved within the confines of a semantic constituent parser. This can sometimes lead to problems, as with the phrase "flight and NOR hours". One constituent parser parses "flight", and a semantic interpretation is made of its meaning only on the basis of that one word. It is interpreted as indicating number of flights (in this domain, the only other choice available to PLANES is flight hours). "NOR hours" is parsed in a separate constituent parser. There is no larger structure to determine that both "flight" and "NOR" modify hours, so "flight hours" was intended.

## 7.4 Accuracy

PLANES assumes that each user's utterance is meaningful, and attempts to build a database query from it. This causes problems because not every utterance is meaningful as a database query. PLANES often answers questions that were not asked or intended. Also, because PLANES discards the sentence level structure of the utterance, utterances with complex structures are often misinterpreted. Data is returned to the

user, but it may not be the data that he requested.

## 7.5 Partial Understanding

Because PLANES employs a top-down parser that relies on backtracking for handling nondeterminism, it can give little assistance to the user when an utterance is not understood. It cannot indicate which phrases have been parsed and are considered correct. If, however, the top level were implemented as a bottom-up parser, all the recognized constituents could be made available to the user. This would give him specific help in determining what the system understood and where it went wrong. The user could then rephrase his utterance accordingly. Without this information, the user must generate a rephrasing without much direction as to where the difficulty lies.

## 7.6 Pragmatics Of Dialog

The current implementation of PLANES does not generate the kind of helpful indirect responses that Kaplan [1979] or Siklossy [1977] describe. However, the new query language being implemented by Gary Brooks keeps track of which predicates (if any) are never satisfied, hence the kind of responses that Kaplan describes could be handled (and handled more efficiently than Kaplan suggested.)

## 7.7 Restricted Subdialogs

PLANES has two forms of restricted subdialogs. One is clarification dialogs that are used when words are not recognized, or when the referent for a pronoun cannot be identified. The other is a help dialog that is a CAI-type frame display (modeled after the PLATO format) that gives the user information about PLANES and the database. The form of interaction is primarily menu selection. The help system was disabled during the testing.

## 7.8 Portability

PLANES, like any semantic grammar system, puts a heavy burden on the implementor to produce a system that is consistent and complete. It was written to be compatible with relational databases, accessed through the query language written for PLANES. It has not been implemented in any other domains.

There are some domains which would not be amenable to a natural language interface like PLANES that discards the top level structure of sentences. For example, any domain which requires directional relationships (John sued Mary, Mary slapped John) could not be handled by PLANES.

## 7.9 Interaction Time

Parsing and retrieval times were discussed early in this chapter. One observation is that the processing is being done at the time when it should be minimized, while the user is awaiting an answer. This is not just a problem with PLANES, but is true of all current natural language systems. The user thinks, enters his query and waits for a response. A great deal of time is spent in the thinking and entering phase. This time could be better spent by the natural language processor in drawing inferences, reorganizing memory to improve access time, and analyzing the utterance as it is generated. This could reduce the perceived response time from tens of seconds to only a few seconds.

## 7.10 Operating Cost

With a database as large as the one used with PLANES, the user is in danger of unwittingly asking a question that could require an enormous amount of searching. If the user is assumed not to be aware of the structure of the database, he cannot be expected to avoid this kind of query. It becomes imperative that the system, through some means, keep the user informed as to the expected cost of the request that he has made, and the expected amount of processing that remains before the search is complete (one hates to terminate a search after several minutes: he never knows whether the answer will

be displayed within seconds, or still take many more minutes.)

## 7.11 Design Of The Environment

The language processing task can be aided by proper design of the interaction environment. For example, if the display were paged so that data would not inadvertently scroll off the top of the screen, there is little use to specifying quotas. Similarly, if a special character is used to terminate user utterances, the system is not presented with the difficulty of deciding whether the user has completed what he intended to say, or he is about to add another sentence, qualifying the thought further. PLANES uses the carriage return character in this way, but has the problem then of accommodating user utterances that exceed one screen line in length.

One comment that was made by several users was that it would be helpful to know how long a query would take, and it would be helpful to let the longer ones continue execution in background mode, while interaction continues with shorter queries in the foreground.

# Chapter 6

# CONCLUSIONS

I have described the nearly total lack of evaluation in natural language processing research. Natural language processing systems have been described only in terms of inputs that are appropriately handled. There generally is no attempt to describe the limitations of natural language processors. This bias toward only illustrating the positive aspects of a system leaves several crucial questions about the system unanswered: what has been accomplished, what problems remain, how general are the solutions to language understanding problems, how do the proposed solutions compare to the solutions proposed by others.

The lack of evaluation and the lack of thorough description that this implies impedes the development of the field. It is impossible to make a reasoned decision on what

work should be continued, extended or abandonned based on the kinds of system descriptions that are currently being presented.

A condition that adds to the evaluation problem is that the goals of natural language processing research are not always clearly stated. They tend to differ significantly between researchers. They are usually only vaguely implied by the work rather than explicitly enumerated. Vague goals makes evaluation more difficult. Different goals between researchers make systems incompatable for comparison. However, if an evaluation can do nothing else, it can make the goals of a research project explicit and clear, and describe the achievements of a particular project against those goals. This would be a significant improvement over current practice.

Current practice for natural language processor evaluation and description is poor at best. The primary techniques used are descriptions of programs and lists of successful examples. Virtually no near-miss examples are ever presented. Dialogs with actual users are almost never given. Controlled experiments exposing systems to naive users are rare. In addition, there has not been an evaluation technique or set of techniques that will produce an adequate description of natural language systems. This thesis is an attempt to describe such an evaluation technique and to apply it to one natural language processing system.

The problem of evaluating a natural language processor is a difficult one. This is primarily due to two factors. First, there is a large number of facts or rules involved in language understanding. It difficult to try to characterize and try to compare different bodies of rules. This makes a collective approach, such as a statistical study of performance, more attractive. However, that is the second difficulty. Statistical studies are corrupted by several factors: the knowledge that users have about the capabilities of the system, the knowledge that users have about the domain, the quality of the implementation, the nature of the problems that users are given to solve and the dependence of the formalism on the domain of discourse.

I feel that if we are to perform meaningful and useful evaluations of natural language processors, we must deal with the problem of their involving many facts. Describing systems in brief journal articles is not sufficient to convey usefully detailed characterizations of the capabilities of systems. I have stressed this in the evaluation technique described in this thesis. The various taxonomies of issues presented in this thesis are an attempt to organize the large body of facts common to many natural language processors.

I have described different points of view for considering a natural language processor. These are an attempt to understand the capabilities of the formalism. The

habitability view considers the natural language processor from the point of view of the user. It is intended to reveal how well the system does what it was designed to do. The completeness view considers the natural language processor from the point of view of the user interacting with an ideal system. The completeness view is intended to reveal how users would interact with an ideal system, i.e., was the system designed to do what the users really wanted it to do. The abstract view considers issues that are not directly observable by the user: how the system was implemented, whether it can be used in other domains of discourse, and what regularities of language have been captured in the formalism.

I have also described the performance of PLANES in more detail than it has previously been described. The evaluation of PLANES is valuable in its own right, but also serves as an illustration of the evaluation technique described in this thesis.

The evaluation technique described here is the result of several trial techniques, all of which were found wanting. I do not consider the current evaluation technique to be flawless, but it is better than the other techniques that I tried, and clearly better than the prevailing technique -- no evaluation at all.

The current method is primarily a guide for careful description of a natural language processor. There are two important desiderata that it does not meet. First, it does not reduce the evaluation to a compact score. After considering the evaluation of natural language processors for some time, I do not feel that such a score can be meaningful except for comparison purposes for specific applications. Second, the technique presented here does not replace informed judgement with objective measurement. The evaluation method requires careful judgements to be made by an evaluator thoroughly familiar with the system he is evaluating. On the positive side, this evaluation method is more thorough than any other, and it is systematic. I feel that an evaluation method with these two characteristics is the contribution of this thesis.

# REFERENCES

Berwick, R. C. "Learning Structural Descriptions of Grammar Rules from Examples," in Proceedings of the International Conference on Artificial Intelligence, 1979.

Bobrow, D., R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson and T. Winograd. "GUS, A Frame-Driven Dialog System." Artificial Intelligence. (1977)8:155-173.

Brown, J. S. and R. R. Burton. "Multiple Representations of Knowledge for Tutorial Reasoning" in Representation of Understanding. D. G. Bobrow and A. Collins. New York: Academic Press, 1975.

Brown, J. S., R. R. Burton and A. G. Bell. "SOPHIE/A Sophisticated Instuctional Environment for Teaching Electronic Troubleshooting." Report 2790, Al Report 12. Cambridge, Mass.: Bolt, Beranek and Newman,1974.

Bullwinkle, C. "Levels of Complexity in Discourse for Anaphora Disambiguation and Speech Act Interpretation" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Burton, R. R. "Semantic Grammar: An Engineering Technique for Constructing Natural Language Understanding Systems." Report 3453, ICAI Report 3. Cambridge, Mass.: Bolt, Beranek and Newman, 1976.

Clark, H. H. and E. V. Clark. Psychology and Language. New York: Harcourt, Brace and Jovanovich, Inc., 1977.

Codd, E. F. "Seven Steps to Rendezvous with the Casual User" in Data base Mangement. J. W. Klimbie and K. L. Koffema, eds. New York: North Holland Publishing Company, 1974.

Codd, E. F., R. S. Arnold, J. M. Cadiou, C. L. Chang and N. Roussopoulos. "Rendezvous Version 1: An Experimental English-Language Query Formulation System for Casual Users." Report RJ2144(29407). San Jose, Calif.: IBM Research Laboratory, 1978.

Damerau, F. "Advantages of a Transformational Grammar for Question Answering" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge Mass.: MIT, 1977.

Damerau, F. "The Derivation of Answers from Logical Forms in a Question Answering System." Research Report RC 6859

( 29411). Yorktown Heights, New York: IBM, Sept. 1977.

DeJong, G. "Prediction and Substantiation: Two Processes that Comprise Understanding," in Proceedings of the International Conference on Artificial Intelligence, 1979.

Finin, T., B. Goodman, H. Tennant. "JETS: Acheiving Completeness through Coverage and Closure," in Proceedings of the International Conference on Artificial Intelligence, 1979.

Gershman, A. V. "Conceptual Analysis of Noun Groups in English." Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Ginsparg, J. M. "Natural Language Processing in an Automatic Programming Domain." PhD thesis, Stanford University, Memo AIM-316, Computer Science Department Report STAN-cs-78-671, June 1978.

Grosz, B. J. "The Representation and Use of Focus in Dialogue Understanding." Ph.D. Thesis, University of California, Berkeley, 1977.

_____ "The Representation of Use of Focus in a System for Understanding Dialogs" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Harris, L. R. "Experience with ROBOT in 12 Commercial Natural Language Data Base Query Applications," in Proceedings of the International Conference on Artificial Intelligence, 1979.

_____ "User Oriented Data Base Query with the Robot Natural Language Query System." International Journal of Man-Machine Studies. (1977)9:697-713.

Heidorn, G. E. "Automatic Programming Through Natural Language Dialogue: A Survey." IBM Journal of Research and Development, 20(4), July, 1976.

Hendrix, G. G. "Human Engineering for Applied Natural Language Processing" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

_____ "LIFER: A Natural Language Interface Facility." SRI Technical Note 135, Dec., 1976.

Hendrix, G. G., E. D. Sacerdoti, D. Sagalowicz and J. Slocum. "Developing a Natural Language Interface to Complex Data" in ACM Transactions on Database Systems, 1978.

Hobbs, J. "Pronoun Resolution." Research Report 76-1. New York, City College, 1976.

____ "38 Examples of Elusive Antecedents from Published Texts." Research Report 77-2. New York: City College, 1977.

Joshi, A. K, S. J. Kaplan and R. M. Lee. "Approximate Responses from a Data Base Query System: An Application of Inferencing in Natural Language" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Kaplan, S. J. "Cooperative Responses from a Portable Natural Language Data Base Query System." Ph.D. Thesis, University of Pennsylvania, July, 1979.

_____ "Indirect Responses to Loaded Questions" in Proceedings of the Second Workshop on Theoretical Issues in Natural Language Processing. Urbana, Illinois: University of Illinois, 1978a.

_____ "On the Difference Between Natural Language and High Level Query Languages" in Proceedings of the ACM 78. Washington, D. C.: ACM, 1978b.

Kaplan, S. J. and A. K. Joshi. "Cooperative Responses: An Application of Discourse Inference to Database Query Systems" in Proceedings of the Second Annual Conference of the Canadian Society for Computational Studies of Intelligence. Toronto, Ontario: 1978.

Krausse, J. "Preliminary Results of a User Study with the 'User Specialty Languages' System and Consequences for the Architecture of Natural Language Interfaces." IBM Heidelberg Scientific Center, TR 79.04.003, May, 1979.

Lehman, H. "Interpretation of Natural Language in an Information System," in IBM Journal of Researh and Development, 22(5), Sept., 1978.

Lehman, H., N. Ott, M. Zoeppritz. "User Experiments with Natural Language for Data Base Access," in Proceedings of the 7th International Conference on Computational Linguistics, 1978.

Malhotra, A. "Design Criteria for a Knowledge-Based English Language System for Management: An Experimental Analysis." Ph.D. Thesis, MIT, MAC TR-146, 1975.

_____ "Knowledge-Based English Language Systems for Management Support: An Analysis of Requirements," in Proceedings of the International Conference on Artificial Intelligence, 1977.

Mann, W. C. "Improving Methodology in Natural Language Processing" in Theoretical Issues in Natural Language Processing. Cambridge, Mass.: Bolt Beranek and Newman Inc., June 1975.

Mann, W. C., J. A. Moore and J. A. Levin. "A Comprehensive Model for Human Dialogue" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Marcus, M. P. "A Theory of Syntactic Recognition for Natural Languages." PhD Thesis, MIT, Feb., 1978b.

McDermott, D. "Artificial Intelligence Meets Natural Stupidity" in SIGART Newsletter, April, 1976, p. 4.

Michaelis, P. R. "Word Usage in Interactive Dialog with Restricted and Unrestricted Vocabularies." IEEE Transactions on Professional Communication, PC-20(4), Dec., 1977.

Miller, H. G., R. L. Hershman and R. T. Kelly. "Performance of a Natural Language Query System in a Simulated Command Control Environment." Advanced Command and Control Architectural Testbed, Code 832. San Diego, Calif.: Naval Ocean Systems Center, 1978.

NAILSC. "NALDA: System Data Requirements Determination Report." Naval Aviation Integrated Logistic Support Center, 1974.

Nash-Webber, B. and R. Reiter. "Anaphora and Logical Form: On Formal Meaning Representation for Natural Language" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Petrick, S. R. "On Natural Language Based Computer Systems." IBM Journal of Research and Development. (July 1976)314-325.

Quirk, R., S. Greenbaum, G. Leech, J. Svartik. A Grammar of Contemporary English. London: Longman Group Ltd, 1972.

Sacerdoti, E. D. "Language Access to Distributed Data with Error Recovery" in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Searle, J. Speech Acts. Cambridge, Mass.: University Press, 1970.

Sheridan, P. B. "On Dealing with Quantification in Natural Language Utterances." International Journal of Man-Machine Studies. (1978)10:367-394.

Sidner, C. "A Progress Report on the Discourse and Reference Component of the PAL." Cambridge, Mass.: MIT Artificial Intelligence Laboratory, 1978.

_____ "Towards a Computational Theory of Definite Anaphora Comprehension in English Discourse." Ph.D. thesis, MIT Artificial Intelligence Laboratory, 1979.

Siklossy, L. "Question-Asking Question-Answering." Report TR-71. Austin: University of Texas Department of Computer Sciences, 1977.

Tennant, H. R. "Experience With the Evaluation of Natural Language Question Answerers" in Proceedings of the International Joint Conference on Artificial Intelligence. 1979.

_____ "The Automatic Advisor." M. S. Thesis, University of Illinois at Chicago Circle, 1977.

Thompson, F. B., B. H. Thompson. "Practical Natural Language Processing: The REL System as Prototype" in Advances in Computers, M. Yovits and M. Rubinoff (eds.). vol. 13. New York: Academic Press, 1975.

VanLehn, K. A. "Determining the Scope of English Quantifiers." AI-tr-483. Cambridge, Mass.: MIT Artificial Intelligence Laboratory, 1978.

Waltz, D. L. "Natural Language Access to a Large Data Base" in Advance Papers of the Fourth International Joint Confeence on Artificial Intelligence Cambridge, Mass.: MIT, 1975

Waltz, D. L. "An English Language Question Answering System for a Large Relational Database." CACM, 21(7), July, 1978.

Waltz, D. L. and B. A. Goodman. "Writing a Natural Language Data Base System." in Proceedings of the International Joint Conference on Artificial Intelligence. Cambridge, Mass.: MIT, 1977.

Watt, W. C. "Habitability." American Documentations. (July 1968) 338-351.

Webber, B. S. "Description Formation and Discourse Model Synthesis" in Proceedings of Theoretical Issues in Natural Language Processing--2. Urbana, Illinois: Univ. of Illinois, 1978a.

_____ "Discourse Model Synthesis Preliminaries to Reference." Cambridge Mass: Bolt Beranek and Newman Inc., 1978b.

_____ "A Formal Approach to Discourse Anaphora." Report 3761. Cambridge, Mass.:Bolt Beranek and Newman Inc., 1978c.

Winograd, T. Understanding Natural Language. New York: Academic Press, 1972.

Wilks, Y. "Methodology in AI and Natural Language Understanding" in Theoretical Issues in Natural Language Processing. Cambridge, Mass: Bolt Beranek and Newman Inc., June 1975.

Woods, W. A. "A Personal View of Natural Language Understanding" in SIGART Newsletter, no. 61, Feb., 1977.

_____ "Procedural Semantics for Question-Answering Machine" in Proceedings of the Fall Joint Computer Conference. Montvale, N. J.: AFIPS Press, 1968.

_____ "Progress in Natural Language Understanding--An Aplication to Lunar Geology" in Proceedings of the National Computer Conference. Montvale, N. J.: AFIPS Press, 1973.

_____ "Semantics and Quantification in Natural Language Question Answering." Report 3687. Cambridge, Mass: Bolt, Berenak and Newman, 1977.

_____ "Semantics for a Question Answering System." PhD thesis, Division of Engineering and Applied Physics, Harvard University, 1967. (also Report NSF-19 Harvard Computation Laboratory.)

_____ "Some Methodological Issues in Natural Language Understanding" in Theoretical Issues in Natural Language Processing. Cambridge, Mass.: Bolt Beranek and Newman Inc., June 1975.

Woods, W. A., R. M. Kaplan and B. Nash-Webber. The Lunar Sciences Natural Language Information System: Final Report. Report 2378. Cambridge Mass.: Bolt Beranek and Newman, 1972.

# APPENDIX

## Test Materials and User Utterances

## Introduction to PLANES Database Given to All Users

### Description of the PLANES Data Base

The PLANES data base contains information on 38 naval aircraft, collected between 1968 and 1972. This information includes maintenance and flight data. Some of the information is stored in alphanumeric codes. Although encoding and decoding may not need to be done by the problem designers or system users, the codes can be found in the Work Unit Code Manuals for A7 and F4 aircraft. The following describes the data types in the system.

1.0 GENERAL

Aircraft: There are two aircraft types in the system, A7's and F4's. Each aircraft is assigned a unique serial number (called BUSER, for bureau/serial number). For simplicity, all BUSER numbers in PLANES are between 1 and 50. Each aircraft is assigned to a permanent unit (PUC). At any particular time the aircraft may or may not be located at its assigned permanent unit. An aircraft may be away from its pemanent unit when it is on an aircraft carrier. If an aircraft spends time on an aircraft carrier, that time is recorded in hours.

An aircraft can be either fully equipped or not fully equipped. It can also be in one of three conditions: (1) operationally ready; (2) not operationally ready (NOR) in which the aircraft cannot be used to perform any of its primary missions; (3) reduced mission capability (RMC) in which the aircraft is capable of safely performing one or more of its primary missions, but not all. There are several reasons for not operationally ready (NOR) and reduced mission capability (RMC) conditions, such as:

Not Operationally Ready (NOR)

1. Not operationally ready because the aircraft was having maintenance performed (NORM). This maintenance can be either scheduled (NORMS) or unscheduled (NORMU).
2. Not operationally ready because the aircraft is waiting for a part that has been ordered from supply (NORS).

Reduced Mission Capability (RMC)

1. Reduced mission capability due to maintenance, either scheduled (RMCMS) or unscheduled (RMCMU).
2. Reduced mission capability due to being not fully equipped (RMCNFE).

If an aircraft has NOR or RMC time during a given day, the starting and ending times are recorded, along with the time span. Also recorded are the part or system that caused the NOR or RMC time, the date and the number

of hours from the discovery of the problem until the maintenance was performed (AWM or awaiting maintenance time).

Parts: Each part that is installed or removed from an aircraft has a manufacturer, a part number, and a serial number. When certain parts are installed or removed, the reading on an aircraft meter is noted (the meter is a timer that runs while the aircraft is in use). This meter time is used to determine the service life of removed parts.

## 2.0 MAINTENANCE

Each maintenance job performed on an aircraft is assigned a unique job control number (JCN). In addition, the following information is also recorded:

1. Aircraft identification.
2. The date that maintenance action was taken (ACTDATE).
3. The organization that performed the action (ACTORG).
4. The work center at which it was performed (ACTWC).
5. A designation of which system, subsystem, component and part were being worked on (WUC).

6. A designation of what action has been taken (AT).
7. Whether the maintenance was scheduled or unscheduled.
8. The type of malfunction for repaired parts (HOWMAL).
9. When the malfunction was discovered.
10. The quantity of parts of a given part number that were removed.
11. The number of manhours expended on the job (MANHRS).
12. The number of clock hours expended on the job.
13. NOR or RMC hours due to the maintenance action.
14. The time maintenance was begun (BEGINTIME).
15. The time maintenance was completed (ENDTIME).
16. The time the aircraft sat idle awaiting maintenance (AWM).

## 3.0 FLIGHTS

The following data is recorded about each aircraft for every flight it takes:

1. Aircraft identification.
2. Date (ACTDATE).
3. Departure time (DEPTIME).
4. Arrival time (ARRTIME).

5. Purpose of flight (FPC).
6. Number of flighthours (FLTHRS).
7. Landing codes:
    1. day or night
    2. ship landings:
        1. arrested or not arrested
        2. touch and go
    3. land/feild landings:
        1. arrested or not arrested
    4. ditched on land or water
8. Catapult flight.

## Instructions for Habitability Users

---

### Instructions to PLANES Test Users

You will be given a list of database search problems that you are to solve using the PLANES system. Before you start using PLANES, study the description of the database to become familiar with it. When you feel that you have an understanding of the content of the database, solve the problems that you have been given in the order they are presented. Try to find full and complete answers to each problem, but if you are having great difficulty making PLANES understand you, you may proceed without finishing a problem. If you wish to return to a problem that was left unfinished, you may do so after you have attempted all the problems.

Habitability Test Problems

---

Subject:

Problem [33]

Please fill in the following table for June, 1972.

| BUSER NO. | NOR HOURS | FLIGHT HOURS |
|---|---|---|
| 13 | | |
| 14 | | |
| 15 | | |
| 20 | | |
| 22 | | |
| 25 | | |

Subject:

Problem [41]

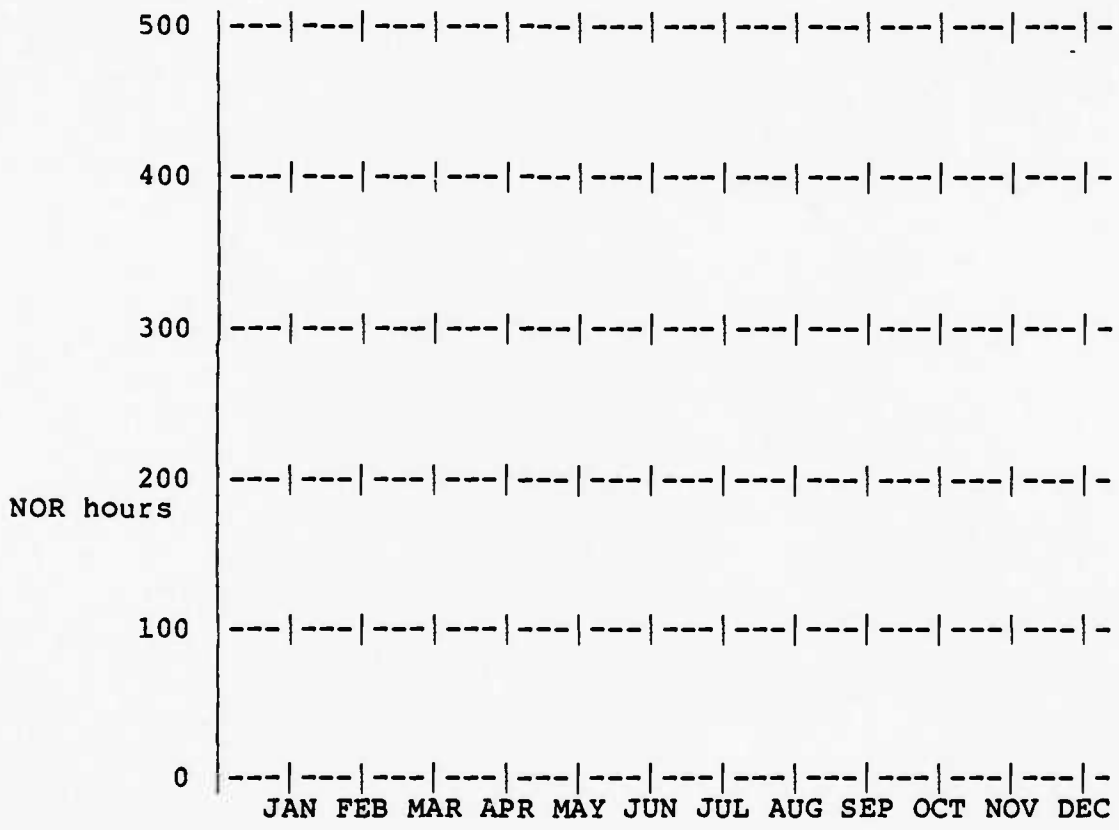
Please fill in the following table for July, 1972.

| | no. of F4 aircraft | no. of A7 aircraft |
|---|---|---|
| NOR | | |
| RMCNFE | | |

Subject:

Problem [25]

Please fill in the following graph for plane 4 in 1973.



Subject:

Problem [39]

Please complete the following table for plane 3 during the period June 1 to June 7, 1973.

| Work Center (ACTWC) | Manhours |
| --- | --- |

Subject:

Problem [45]

Please fill in the following table for plane 4 for June
first to seventh, 1973.

| HOWMAL CODE | MANHOURS | ACTION TAKEN (AT) CODES |
|---|---|---|

Subject:

Problem [23]

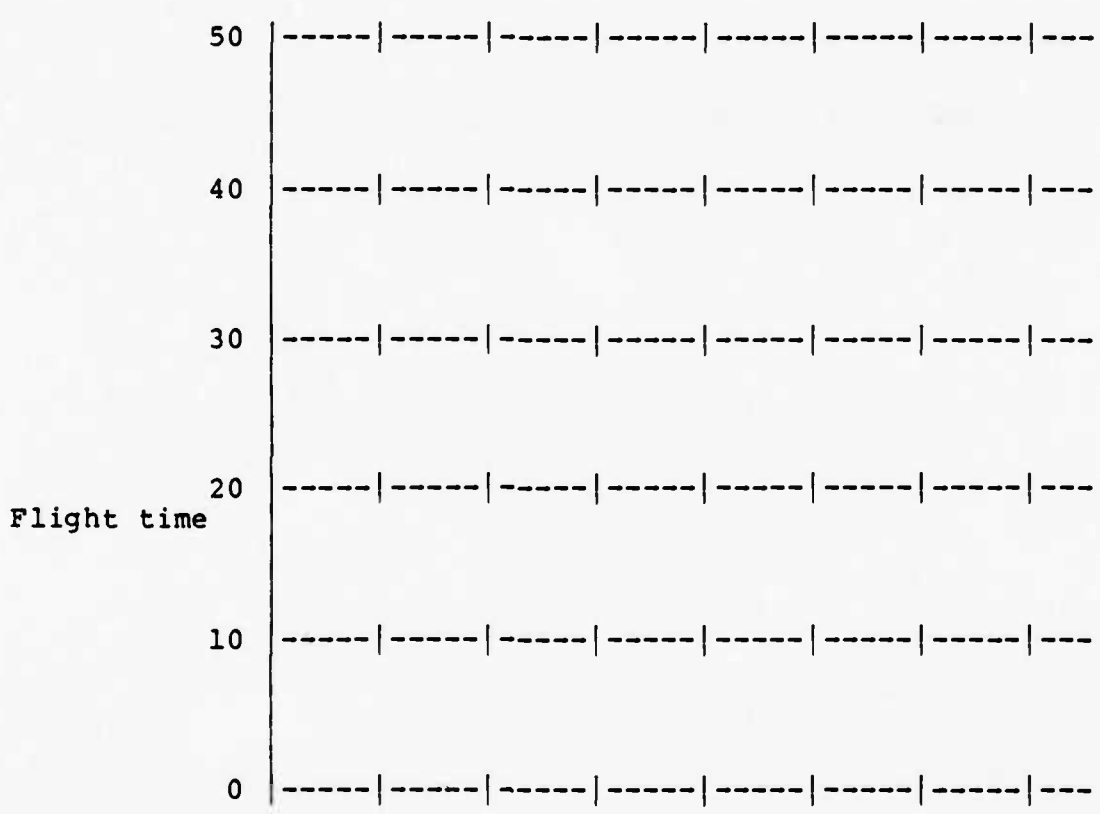
Please fill in the following table for plane 4 in June 26 to June 30, 1973.

| Time taken for flight | flight purpose code (FPC) |
| --- | --- |

Subject:

Problem [52]

Complete the following histogram for December, 1970.



Utterances from Habitability Tests

---

>> list the NOR hours in each month of 1973 for plane 4

>> list the HOWMAL CODE, MANHRS, and AT for each JCN on plane 4 from June first to June seventh, 1973.

>> list the HOWMAL CODE, manhours needed, AT CODES for plane 4 from Junel to 7 in 1973.

>> tell me the HOWMAL CODE, manhours needed, action taken for all maintenance on plane 4 from June 1 to June 7, in1973.

>> tell me FLTHRS, and FPC for all flights in plane 4 from June 26 through June 30, 1973.

>> tell me the NOR, flighthours in June 1972 for plane 13,14,15,20,22,25.

>> tell me the NOR hours, flighthours in June, 1972 for plane 13, plane 14, plane 15, plane 20, plane 22, plane 25.

>> tell me how many manhours were needed in each ACTWC for plane 3 from June 1 to June 7, 1973.

>> tell me the total flighthours in December, 1970 of all planes that flew

>> tell me the flighthours for each plane in December, 1970

>> for July, 1972 tell me how many F4 aircraft were NOR, RMCNFE

>> during July 1972, how many F4 aircraft were NOR

>> during July, 1972 how many F4 aircraft were RMCNFE

>> tell me how many F4 aircraft were NOR, how many F4 aircraft were RMCNFE, how many A7 aircraft were NOR, how many A7 aircraft were RMCNFE during July, 1972

>> list planes NOR during July, 1972

>> how many A7 aircraft were NOR during July,1972

>> how many F4 aircraft were NOR during July, 1972

>> how many F4 aircraft were RMCNFE during July, 1972

>> how man A7 aircraft were RMCNFE during July, 1972

>> how many A7 aircraft were RMCNFE during July, 1972

>> What is the number of NOR F4 aircraft?

>> What is the number of NOR A7 aircraft, and RMCFE F4 and A7 aircraft.

>> List the number of NOR A7 aircraft ,then list the number of RMCNFE F4, and A7 aircraft.

>> What is the number of NOR A7 aircraft?

>> What is the number of RMCNFE F4 and What is the number of RCNFE A7 aircraft

>> How many F4 and A7 aircraft are RMCNFE?

>> List NOR hours for plane 4 in 1973 . State the NOR hours for each month.

>> Find ACTWC and MANHRS BEGINTIME June 1 to June 7, 1973 for plane 3

>> Which ACTWC and for how many MANHRS was plane 3 from begintime June 1 to endtime June 7.

>> What is the ACTWC for plane 3 begintime june 1 to June 7, and what is the number of manhours at each actwc.

>> List nor, flthrs for planes 13, 14, 15, 20, 22

>> How many nor hours and flight hours for planes 13, 14, 15, 20, 22, 25, for June 1972?

>> List seperately nor, flighthours for planes 13, 14, 15, 20, 22, 25 for Junel972.

>> List nor hours for plane 13, 14, 15, 20, 22, 25 for June 1972.

>> How many flight hours for plane 13, 14, 15, 20, 22, 25 for June 1972?

>> List nor hours for each plane 13, 14, 15, 20, 22, 25, for June, 1972.

>> List the flight hours for each plane 13, 14, 15, 20, 22, 25, for June 1972.

>> List all planes having flight time for December, 1970.

>> what are the flight times for all the aircraft in december,1970

>> what is the flight time for all the aircraft for december, 1970

>> list the flight times for december, 1970

>> list the flight times for aircraft 1 through plane 50 for december, 1970

>> list the total flight times for december, 1970

>> what is the nor hours for plane 4 in 1973

>> what is the howmal code, manhours, and at code for plane 4 from june 1 to june 7, 1973

>> how many aaf had nor for july, 1972

>> how many nor did f4 and a7 have for july, 1972

>> how many nor are there for f4 for july, 1972

>> how many nor for a7 for july, 1972

>> how many f4 had rmcnfe for july, 1972, how many a7 had rmcfe for july, 1972

>> how many rmcnfe are for f4 for july, 1972

>> what is the total number of rmcnfe for a7 for july, 1972

>> how many f4 had rmcnfe for july, 1972

>> what fpc and flight time did plane 4 have for june 26, 1973 to june 30, 1973

>> what was the flight hours for each fpc for plane 4 in june 26, 1973 to june 30, 1973

>> list the flight hours for each fpc for plane 4 in june 26, 1973 to june 30, 1973

>> list the flight hours for plane 4 in june 26, 1973 to june 30, 1973

>> what are the flight hours for each fpc for plane 4 for june 26, 1973 to june 30, 1973

>> what actwc and how many manhours are for plane 3 for june 1, 1973 to june 7, 1973

>> what are the nor and flight hours for plane 13, 14, 15, 20, 22, 25 for june, 1972

>> what is the flight hours for plane 1 to plane 50 for december, 1970

>> what plane have flight hours for december, 1970

>> how many hours was plane 4 down in january of 1973.

>> how many hours was plane 4 NOR in feb 1973, and mar 1973

>> how many hours was plane 4 nor in feb, mar, apr in 1973

>> list how many hours busar 4 was nor for each month in 1973

>> how many hours was buser 4 nor in feb 1973

>> list for the months of march, april, may, june, july, august, september, october, november, and december of 1973 that buser 4 was nor

>> list for each month in 1973 that buser 4 was nor

>> how many hours was buser 4 nor in mar 1973

>> how many hours was buser 4 nor in apr 1973

>> list nor hours for buser 4 for each month in 1973

>> list for buser 4 the number of flight hours for each day from june 26 1973 to june 30 1973

>> how much time taken for flight did buser 4 have on june 26 1973

>> request the number of flight hours for buser 4 on june 26 1973

>> how many flight hours did buser 4 have on june 26 1973

>> how many flight hours did buser 4 have on each day of june between the 26 and the 30 of june

>> how many flight hours did buser 4 have on june 27 1973

>> how many flight hours did buser 4 have on june 28 1973

>> list for june 28th to june 30th the number of flight hours for buser 4

>> list for buser 4 the number of flight hours on june 29 and 30 of 1973

>> how many flight hours did buser 4 have on june 29 and june 30 of 1973

>> how many flight hours did buser 4 have on june 29 1973

>> how much flight time did buser 4 have on june 30 1973

>> how many flight hours did buser 4 have on june 30 1973

>> list the fpc for buser 4 for june 26 to june 30

>> how many f4 aircraft were nor for july 1972

>> how many f4 aircraft were rmcnfe in july 1972

>> how many a7 aircraft were nor in july 1972

>> how many a7 were nor in july 1972

>> how many a7 were rmcnfe in july in 1972

>> how many a7 were rmcnfe in july 1972

>> how much nor time did buser 13 have in june 1972

>> how much nor time did buser 14 have in june 1972

>> how much nor time did buser 15 have in june 1972

>> how many nonoperational f4 aircr ft are there

>> how many nor f4 aircraft are there

>> how many rmcnfe f4 aircraft are there

>> how many nor and rmcnfee a7 aircraft aare

>> how many nor hours did plane 4 have in january 1973

>> how many nor hours did plane 4 have in the consectutive months of 1973

>> how many nor hours did plane 4 have in the month of february 1973

>> how many nor hours did plane 4 have in march 1973

>> how many nor hours did plane 4 have in april may june july august septemberoctober november and december seperately

>> how many nor hours did plane4 have in april 1973

>> how many nor hours did plane 4 have in may 1973

>> how many nor hours did plane 4 have in june 1973

>> how many nor hours did plane 4 have in july 1973

>> how many nor hours did plane 4 have in august 1973

>> how many nor hours did plane 4 have in september 1973

>> how many nor hours did plane 4 have in october 1973

>> how many nor hours did plane 4 have in november 1973

>> how many nor hours did plane 4 have in december 1973

>> what is the time taken for flight from june 26 to june 30 1973

>> how much time taken for flight for plane 4 from june 26 to june 30 1973

>> how much has plane 4 flown from june 26 to june 30 1973

>> how many hours did plane 4 fly from june 26 to june 30 1973

>> how many hours has plane 4 flown from june 26 to june 30 1973

>> what was fpc for plane 4 from june 26 to june 30 1973

>> what are nor hours for buser 13 for june 1972

>> how many flight hours did busser 13 have in june 1972

>> how many nor hours did buser 14 have in june 1972

>> how many flight hours did buser 14 have in june 1972

>>how many nor hours did buser 15 have in june 1972

>>how many flight hours did buser 15 have in june 1972

>> how many nor hours did buser 20 have in june 1972

>> how many flight hours did buser 20 have june 1972

>> how many nor hours did buser 22 have in june 1972

>> how many flight hours did buser 22 have in june 1972

>> how many nor hours did buser 25 have in june 1972

>> how many flight hours did buser 25 have in june 1972

>> what was actwc for plane 3 on june 1 1973

>> what were manhours for plane 3 on june 1 1973

>> what was howmal code for plane 4 for june first 1973

>> what was the purpose of the flight of plane 4 from june26 to june 30 1973

>> what was fpc of plane 4 from june 26 to june 30 1973

>> what was actwc for plane 3 from june 1 to june 7 1973

>> what are manhours for plane 3 from june 1 to june 7 1973

>> what is howmal code for plane 4 for june 1 to june 7 1973

>> what are manhours for plane 4 for june 1 to june 7 1973

>> what was at on plane 4 for june 1 to june 7 1973

>> where was airplane 3 during the period June 1 to June 7, 1973?

>> Was airplane 3 NOR during the period June 1 to june , 1973?

>> What was the JCN for airplane 3 during the period June 1 to June 7, 1973?

>> Find the CN assigned to airplane 3 during the period June 1 to June 7, 1973.

>> What is the ACTWC where plane 3 was during the period June 1 to June 7, 1973?

>> What was the ACTWC at which plane 3 was during the period June 1 to June 7?

>> What was the number of airplanes NOR for July ,1974?

>> What was the number of f4 aircrafts NOR for July , 1917?

>> how many f4 aircraft were NOR for july 1972?

>> How many A7 aircraft were NOR for july 1972?

>> How many f4 and a7 aircraft were RMCNFE?

>> What were the flight hours of f4 and a7 aircraft for december ,1970?

>> What was the HOwMAL, ManHours , and action taken for plane 4 for june 1 to june 7 ,1973?

>> How many Nor hours and flight hours did planes 13, 14 , 15, 20, 22,and 25 have for june ,1972?

>> How many nor hours and flight hours did each plane 13,14,15 ,20,22,and 25 have for june 1972?

>> how many nor and flight hours did each plane 13,14 ,15, 20,22,25 have for june, 1972?

>> How many nor hours did plane 4 have for each month in 1973?

>> How many nor hours for each month did plane 4 have in 1973?

>> What was the actwc and the manhours for plane 3 from june 1 to june 7, 1973?

>> What was the actwc for plane 3 during the period june 1 to june 7?

>> How many manhours did plane 3 have form june 1 to june 7?

>> How many nor and flight hours did plane 13 have for june 1972?

>> How many flight hours and nor hours did plane 14 have for june 1972?

>> How many flight and nor hours did plane 20 have for june 1972?

>> how many nor and flight hours did plane 22 have for june ,1972?

>> how many nor and flight hours did plane 25 have for june ,1972?

>> How many fligth hours did plane 4 have for june 26 to june 30 ,1973?

>> What was the fpc for plane 4 in june 26 to june 30 ,1973?

>> list flight purpose codes for plane 4 from june 26 to june 30, 1973

>> did plane 4 fly only three missions from june 26 to june 30, 1973

>> list times for flight for plane 4 from june 26 to june 30, 1973

>> list all flights of plane 4 from june 26 to june 30, 1973

>> list fpc and flighthours for plane 4 from june 26 to june

30, 1973

>> what are fpc and flighthours for plane 4 from june 26 to june 30, 1973

>> what busers were active in december, 1970

>> which planes flew in december, 1970

>> which planes flew in december, 1970

>> what was total flighthours for plane 1 in december, 1970

>> what were sum total flighthours for planes 2, 6, 45, and 48 in december, 1970

>> what were the total flighthours of plane 2 in december, 1970

>> what were total flighthours of plane 6 in december 1970

>> what were total flighthours for plane 45 in december 1970

>> what were total flighthours for plane 48 in december 1970

>> did plane 48 fly in december 1970

>> which planes flew in december 1970

>> what were the individual total flighthours for planes 1, 2, 6, 45, 48 in december 1970

>> what was the actwc and manhours for maintenance on plane 3 from june 1 to june 7, 1973

>> is there any nor or rmc time for plane 3 from june 1 to june 7, 1973

>> make table of total nor hours and total flighthours for each of following planes: 13, 14, 15, 20, 22

>> what are nor hours and flighthours for planes 13, 14, 15, 20, 22, 25 for june, 1972

>> what is planetype code for f4 and a7

>> how many f4 aircraft were nor and rmcnfe during july, 1972

>> how many f4 aircraft were nor in july, 1972

>> how many f4 aircraft were rmcnfe in july, 1972

>> how many a7 aircraft were nor and how many were rmcnfe in

july, 1972

>> how many a7 aircraft were nor in july, 1972

>> how many a7 aircraft were rmcnfe in july, 1972

>> what were the nor hours for plane 4 for each month of 1973

>> what were the howmal codes, manhours, and at codes for plane 4 from june 1 to june 7, 1973

>> how many nor hours buser no. 13

>> how many nor hours for buser no. 13 in june 1972

>> how many flight hours for no.13 in june 1972

>> how many nor hours for no. 14 in june 1972

>> how many flight hours for no. 14 in june 1972

>> how many nor hours for no.15 in june 1972

>> how many nor hours for no. 20 in june 1972

>> how many nor hours for no.22 in june1972

>> how many nor hours for no. 25 in june 1972

>> how many nor hours for no. 13 in june 1972

>> how many flight hours for no. 13 in june 1972

>> no. flight hours ,no.15 in june 1972

>> how many flight hours in june 1972 for no.15

>> in june 1972, how many flight hours for no.20

>> how many flight hours for no.22 in june 1972

>> how many flight hours for no.25 in june 1972

>> how many nor f4 aircraft in july 1972

>> how many f4 aircraft nor for july 1972

>> how many f4 aircraft rmcnfe in july 1972

>> how many a7 nor july 1972

>> how many a7 rmcnfe july 1972

>> how many f4 rmcnfe july 1972

>> no.4 how many flights june 26

>> flight

>> find all hours first flight june 26 1973 no 4

>> how many hours flight no.1 june 26 1973 no.4

>> what fpc no.4 june 26 1973

>> what flight hours no4 june 26 1973

>> what fpc no.  4 june 26 1973

>> find all flight hours no.  4 june 27 1973

>> what fpc no.  4 june 27 1973

>> what flight hours no.4 june 28 1973

>> find fpc no.4 june 28 1973

>> what flight hours no.4 june 26 1973

>> find flight hours no.4 june 29 1973

>> what flight hours no.4 6 1973

>> what flight hours no.4 june 30 1973

>> what nor hours per month 1973 no.4

>> what nor no.4 jan 1973

>> find nor no.4 feb 1973

>> find nor mar 1973 no.4

>> find nor 1973 no.4

>> find nor 1973 no.4

>> find nor 1973 months no.4

>> find nor apr 1973 no.4

>> find nor no.4 1973

>> find nor hours 1973 no.4

>> what nor may 1973 no.4

>> list flight hours dec 1970 no.1 2 3 4 5 6 7

>> what aircraft are NOR or RMCNFE in july i972

>> how many4 aircraft were NOR in july 1972

>> how many f4 aircraft were rmcnfe in july 1972

>> how many A7 aircraft were NOR in july 1972

>> how many A7 aircraft were RMCNFE in july 1972

>> how many flight hours did buser no. 13 have for june 1972

>> how many flight hours did aircraft no.14 have in jjune 1972

>> how many NOR hours in june 1972 did each of the following aircraft have;no.1,no.14,no.15,no.20,no.22,and no.25

>> how many flight hours did aircraft no. 20 have in june 1972

>> how many hours did aircraft no. 22 have in june 1972

>> how many flight hours did aircraft no. 25 have in june 1972

>> how many NOR hours did aircraft no. 13 have in june 1972

>> how many NOR hours did aircraft no. 14 have in june 1972

>> how many NOR hours did aircraft no. 15 have in july 1972

>> what was NOR hours for aircraft no.20 in june 1972

>> how many NOR hours did aircraft no. 22 have in june 1972

>> how many NOR hours did aircraft no. 25 have in june 1972

>> list maintenance performed on aircraft no.3 during the period june 1 to june 7 1973

>> what maintenance was performed on aircraft no. 3 during the period june 1 to june 7 1973

>> what was repaired on aircraft no.4 from june 1 to june 7 1973

>> what work was performed on plane no. 4 during june 1 to june 7 1973

>> what type of malfunctions occured on aircraft no.4 from june 1 to june 7 1973

>> what was the nor hours for buser no 13 for june 1972

>> for buser 14,15,20, 22, 25 what the number of nor hours and flight hours for june 1972? what is the number for flight hours for buser 13 for june 1972

>> what is the number of nor hours for june 1972 for buser no. 14, 15, 20, 22, and 25

>> what is the number of nor hours for june 1972 for the aircraft number 14

>> what is the number of flight hours for the aircraft number 13 during june 1972

>> what was the number of flight hours for the aircraft with the numbers 14, 15, 20, 22, 25 for june 1972

>> what was individual number of flight hours during june 1972 for aircraft 14, 15, 22, 20, 25

>> what is the nummber of flight hours for aircraft 14 during june 1972

>> what was number of nor hours for aircraft 14 during june 1972

>> what number of nor hours for aircraft 15 during june 1972

>> list number of flight hours for aircraft 15 during june 1972

>> what is number of nor hours for aircraft number 20 during june 1972

>> what is number of flight hours for aircraft 20 during june 1972

>> what is number of nor hours for aircraft 13 during june 1972

>> under aircraft with number 13 what is the number of flight hours flown during june 1972

>> i need to know the number of flight hours flown during june 1972 for aircraft with number 13

>> want number of flight hours flown by number 13 during june 1972

>> count number of flight hours flown by number 13 during june 1972

>> what is number of flight hours flown by buser no. 13 on june 1972

>> flight hours flown by buser 13 during june 1972

>> ?

>> total nor hours for no. 14 during june 1972

>> number flight hours flown no 14 during june 1972

>> total flight hour of buser no 14 during june 1972

>> total nor hours for no 115 during june 1972

>> total nor hours for no. 15 during june 1972

>> total flight hours for no. 15 during june 1972

>> total nor hours for no. 20 during june 1972

>> total flight hours for no. 20 during june 1972

>> total nor hours for no. 22 during june 1972

>> total flight hours for no. 22 during june 172

>> total flight hours of no. 22 during june 1972

>> total nor hours of no. 25 during june 1972

>> total flight hours of no. 25 during june 1972

>> total no. of f4 aircraft nor for julu 1972

>> total no. of f4 aircraft nor during july 1972

>> total number of f4 aircraft nor during july 1972

>> total number of a7 aircraft nor during july 1972

>> total number of f4 aircraft rmcnfe during july 1972

>> total number of a7 aircraft rmcnfe during july 1972

>> actwc and manhours for plane 3 during june 1 to june 7 1973

>> name actwc that completed work on plane 3 during june 1, 1973 to june 7, 1973

>> what actwc worked on plane 3 during period of june 1, 1973 to june 7, 1973

>> total manhours plane 3 from june 1, 1973 to june 7 1973

>> total manhours on no. 3 from june 1, 1973 to june 7, 1973

>> total nor hours number 3

>> howmal code for number 4 from june 1 to june 7 1973

>> find the no. of F4 aircraft that were NOR in July 1972

>> Find number of A7 aircraft that were NOR in July 1972

>> find number of RMCNFE F4 aircraft for July 1972

>> find the number of F4 aircraft that were RMCNFE in July 1972

>> find the number of A7 aircraft that were RMCNFE in July 1972

>> All work performed on plane 3 from June 1 through June 7, 1973

>> Find all flight time and the purpose of each flight for plane 4 from June 26 through June 30, 1973

>> find all flight data for plane 4 from June 26 through June 30, 1973

>> find flight records for plane 4 from june 26 to june 30, 1973

>> find FPC and flight time for plane 4 from June 26 to June 30, 1973

>> find FPC and FLIGHTHOURS for plane 4 from June 26 to June 30, 1973

>> find NOR hours and FLIGHTHOURS for Buser no. 13 in june 1972

>> find HOWMAL for plane 4 from june 1 to june 7, 1973

>> find HOWMAL, MANHOURS, and ACTORG for plane 4 from june 1 to june 7, 1973

>> find NOR hours for each month in 1973

>> find NOR HOURS and FLIGHT HOURS for planes 13, 14, 15, 20, 21, 22, and 25.

>> find NOR HOURS and FLIGHT HOURS for plane 13 in june, 1973

>> find NOR HOURS and FLIGHT HOURS for BUSER NO. 13 in june, 1972

>> find NOR HOURS and FLIGHT HOURS for 14 in june 1972

>> find NOR HOURS and FLIGHT HOURS for 15, 20, 22, 25 in june 1972

>> find ACTWC and MAN HOURS for plane 3 from june 1 to june 7, 1973

>> find NOR HOURS for plane 4 for each month in 1973.

>> find all aircraft with FLIGHTHOURS in december, 1970

>> what are the nor hours for jan?

>> what are the nor hours for plane 4 in feb of 1973

>> what are the nor hours for plane 4 in mar of 1973?

>> what are the nors hours for plane 4 in apr of 1973?

>> what are the nors hours for plane 4 in may of 1973?

>> what are the nors hours for plane 4 in jun of 1973?

>> what are the nors hpurs for plane 4 in jul of 1973?

>> what are the nors hours for plane 4 in aug of 1973?

>> what are the nors hours for plane 4 in sep of 1973?

>> what are the nors hours for plane 4 in oct of 1973?

>> what are the nors hours for plane 4 in nov of 1973?

>> what are the nors hours for plane 4 in dec of 1973?

>> what are the nor hours and flight hours for buser 13 in june of 1972?

>> what are the nors hours for buser 13 in june of 1973?

>> what are the nor hours and flight hours for buser 14 for june of 1972?

>> what are the nor hours and flight hours for buser 14 in june of 1972?

>> what are the nor hours and flight hours for buser 15 in june of 1972?

>> what are the nor and flight hours for busers 20,22, and 25?

>> what are the nor and flight hours for buser 20 in june of 1972?

>> what are the nor hours and flight hours for buser 22 in june of 1972?

>> what are the nors hours and flightnhours for june of 1972?

>> what aircraft had flight time during december of 1970?

>> flight time for plane 2 in december of 1970?

>> flight time for plane 6 in dec of 1970?

>> flight time for plane 1 in dec of 1970?

>> flight time for plane 48 in dec of 1970?

>> flight time for plane 45 in dec of 1970?

>> time taken for flight in plane 4 on june 26 to june 30 1973

>> what maintenance has been performed on plane 4 from june first to june 7, of 1973.

>> what are the howmal codes for these actions taken

>> what were the man hours required for the actions taken

>> what was the action taken for the different codes

>> what were the nor hours for plane 4 in january of 1973

>> what were the nor hours required for the months february through december of 1973 on plane 4

>> which aircraft have flight time for december of 1970

>> what is the flight time for plane 1, 2, 6, 45, 48, for december of 1970

>> what was the work center for plane 3 during the period from junel to 7, 1973

>> how many manhours were required for plane 3 from june 1 to 7, 1973.

>> was any work performed on plane 3 from june 1 to 7, 1973.

>> what work center was plane 3 located at from june 1 to 7 , 1973.

>> where was plane 3 located from june 1 to 7 , 1973.

>> what are the nor hours for planes 13,14,15,20,22,25

>> what are the nor hours for planes 13, 14, 15, 20, 22, 25.

>> what were the nor hours for planes 13, 14, 15, 20, 22, and 25 in june of 1972.

>> what were the flight hours for planes 13, 14, 15, 20, 22, and 25 for june 1972.

>> what were the times taken for flight for plane 4 from june 26 to june 30, 1973.

>> what were the times of flight for plane 4 from june 26 to june 30, 1973.

>> how long was the dduration of the flights for plane 4 from june 26, to june 30, 1973.

>> what were the durations of the flights for plane 4 from june 26, to june 30,

>> how many flights did plane 4 fly from june 26, to june 30, 1973.

>> how long did plane 4 fly on june 26, 1973.

>> what was the flight purpose code.

>> what were the flight purpose codes for plane 4 from june 26, to june 30, 1973.

>> what was the duration of the flights.

>> how many f4 aircraft were classified as nor in july, 1972.

>> how many a7 aircraft were nor in july 1972.

>> how many a7 aircraft were nor in july, 1972.

>> how many f4 aircraft were rmcnfe in july, 1972.

>> how many a7 aircraft were rmcnfe in july, 1972.

>> what were the manhours for plane 4 from june 1, to june 7, 1973.

## Instructions to Completeness Users

You are to assume the role of an aircraft maintenance manager for the US Navy. You are responsible for keeping about 50 aircraft in a state of readiness. You have just received the following order from your commanding officer.

The aircraft readiness reports show that you had 25% more NOR time per aircraft in 1971 than in 1970, 1972 or 1973. Investigate this situation and report back to me. Find the source of the unusually high NOR time.

Use the PLANES system to gather whatever data you can find. You may wish to take notes.

## The Completeness Dialog Utterances

>>WHAT WAS THE AVERAGE DOWN TIME OF THE PLANES IN 1971?

>>What was the down time of the planes in 1970?

>>What was the major cause for down time?

>>On aircraft buon aircraft Buser 1, how many hours of down time were due to NORMU

>>From Jan.1, 1971 until Dec. 31, 1971?

>>Of these hours, how many were due to engine malfunctions?

>>What engine component failed most often?

>>What was the down time of plane Buser 1 during 1970?

>>Of this total, how much was due to NORMU?

>>What was the average NORMU on the planes in 1971?

>>what were the average NORMU hours in 1969, 1970, 1972, respectively?

>>What does NIL mean?

>>What was the total number of flight hours accumulated by the planes in 1971?

>>What was the total number of flight hours for all the planes in 1971 added together?

>>What was the average flight time for the planes in 1969, 1970, and 1972?

>> What percentage of the flight time in 1969, 1970, 1971, and 1972 were accumualted while on aircraft carrier?

>>what percentage of the NORMU time in 1969, 1970, 1972, were due to engine malfunctions?

>>How many landings were on ship for plane 1 in 191970, 1971, and 1972?

>>What was the average NORS for the planes in 1970, 1971, and 1972?

>>how many hours of NOR were thee for each type plane during the years 1970, 1971, 1972, and 1973?

>>Please show me which planes had the most NOR hours during each year

>>same list for year 1

>>same list for year2

>>tell me the PUC for buser numbers, 34, 30, 29, 13,33,32,28,26,27

>>give me the puc for buser numbers 34, 30, 29, 33, 6, 2, 1

>>what type maintanance was done on each buser in 1971

>>tell me the total numbers of NORM and NORS for the years 70, 71, 72

>>tell me the total number of NORMS and NORMU hours for years 70, 71, 72, and 73

>>relate to me the total number of flight hours for each of the ave years

>>show me the mumber ofNOR hours due to battle damage during each of the years 0,1,2,3

>>yes

>>how many planes in each of the four years?

>>how many work centers are there

>>how many NORM and NORS in 73

>>were the mechanics that performed the maintenence on the aircraft the same during all of the years in question?

>>What is the average down-time of aircraft at each ACTORG? That includes maintenence time and AWM.

>>How many unscheduled maintenence actions (NORMU) were accomplished during the years in Question?

>>Which WUC was worked on the most during these years?

>>past data from 1968 thru 1972

>>NOR time , RMC time for all the planes each of the years

>>during 70 thru 73 , where did most of the down time take place, ACTORG?

>>what is the break down of NORMU versus NORMS time in 72 and

73 as well

>>how many hours in 70 thru 73 were RMC

>>whar were the total mission times, total for 70 thru 73

>>total flight hours for each year , but not each flight or mission

>>what is the break down of HOWMAL time in 71 between the types of malfunction

>>ok, how about the AWM time , total for each year

>>ok, how about operational readyness time , total for each year ,or does he the total flight time and the total nor time mean the same thing??

>>could I have a comparison between the RMC time in 71 and72?

>>hcould I possibily find out which system required the most maintenance, and if not what information could I get as a break down of each of the maintenance items ????

>>in NORMU, in 71, 72 , what was the expected damage total for 303?

>>okWhat was the average total flight time per aircraft in 1971 as compared to the other 3 years

>>How many of the aircraft were in for NORMU in 1971 as compared to the other three years

>>Were the parts that were ordered for the aircraft in 1971 as readily available as in the other three years

>>What was the AWM for the aircraft in 1971 as compared with the other years

>>Was the service life of the removed parts in 1971 shorter on an average than in the other three years

>>What was the total number of man hours expended on maintenance in 1971 as compared to the other three years

>>Yes

>>Were more landings made with arresting equipment in 1971 than in other years

>>Were the flight purposes in 1971 different than other years

>>Yes

>>What parts or systems caused the NOR or RMC time in 1971

>>Was the ACTORG different in 1971 than the other three years

>>What was the quantity of parts of a given part number that were removed

>>no

>>How much time did the aircrafts spend on an aircraft carrier in 1971 as compared to the other three years

>>Of the aircraft based on an aircraft carrier, how many were NOR while on the ship

>>yes

>>Is there a reason that you can tell me of as to why the AWM time in 1971 was so high

>>Yes

>>For some reason the NOR and RMC hours due to maintenance were high, This is the reason for the high AWM time

>>were there more flight hours in 71 than in the other three years?

>>avg per plane

>>Did the number of mechanics stay the same per year? avg?

>>Werw there more carrier flights in 71 than the other years?

>>How many planes were returned to service with no defects found?

>>List for each year?

>>How many planes were returned to service with no defects found for the four year period?

>> interpreting

>>no

>>List the avg AWM time for each year

>> AWM for 71,72 and 73?

>>yes

>>List top ten locations that had performed the most mat.

over the four yrs.

>>by yr

>>Which planea flew more hours in each of the four yrs?

>>yes

>>Of all possible parts, which part was replaced most often in 70?

>>Which organization performed the most work in 70?

>>Which ACTORG performed the most work in 71?

>>What is PE4?

>>Where

>>Where is it located?

>>How much of the mat. performed in 71 was scheduled? nonscheduled?

>>were there more ship landings made in 1971 as compared to the other >>years?

>>was the higher nor from one of the aircraft, or was it from both aircraft combined?

>>what i meant was for example did the a-7 experience more nor time in 1971 vs. the other years, or was the increased nor time due to both the a-7 and the f-4 being down more.

>>Alright, was the time interval between inspections and regular maintenance the same as the other years?

>>Were the aircrafct flown more when they were in a RMC state as compared to the other years? In other words, were they flown when they were not in top condition?

>>I dont think I need all that quite yet... Was the ACTORG the same in 71 as the other years? Were there a bunch of trainees working on the planes?

>>Was there a WUC that showed a marked increase of failures...some component that went bad a lot during 71... maybe a problem during manufacture at the factory?

>>OK, in 1970 there were few ship landings made as compared to 1971 when there was a large increase of sea landings. Although 1972 had more than 71, the jump was realized between 70 and 71. Due to this increase, were there modifications to

the aircraft that became necessary from the higher number of carrier landings that were made in 71. This might explain why the NOR was greatest in 71...and why it went back down after. The planes had the modifications that they needed after that year.

>>I believe that you said that there were 36 different ACTWCS that preformed service on the aircraft so it is safe to assume that the work was the same performance wise.... During 1971, was the FPC much different than other years? Were the aircraft used for missions that would require higher performance; and by doing so increase the wear and tear on the planes.

>>Why dont you give me the flighthours per FPC

>>Yes, I suppose that flights per FPC would be OK.

>>Was the percentage of scheduled vs. unscheduled maintenance the same in 1971 as in other years?

>>Were the aircraft equipped differently during 71... did they carry specic I meant to say special equipment that required more NOR time to keep the aircraft airworthy?

>>Was parts supply having an off year, thereby increasing the NORS time?

>>If it took longer to get the necessary parts in `71, then how does the AWM time compare?

>>Yes, but only if this file search is considerably shorter than the last.

>>Is there any way that I could get some form of a summary of the JCN`s for 1971? Maybe there is a certain problem that kept happening. Or I guess what I mean to say is the JCN`s different in `71 than the other years?

>>The JCN for all the NORMU shop visits during 1971.

>>A listing of the JCNS that were `caused` by NORMU shop visits during `71

>>No, I suppose that would take hours to get....What I was getting at was trying to see if there was a specific maintenance job during 71 that caused the aircraft to spend more time in the shop. There is some reason for the increased NOR time and I cant see what it is at all. The planes were basicly flown the same (except for the increase in carrier landings, but they didn`t seem to have any real affect), and the quality of the aircraft from the factory as well as the quality of the maintenance preformed seem to be constant. The

parts supply condition probably played some part, but I don`t think that is entirely responsible for the 25% increase of NOR. I don`t really know; any clues?

>>OK, give me a NOR on the `typical` A/C for a year

>>Well, that really didn`t tell me anything as I allready know that the NOR value is the greatest during 1971. Is there one JCN, or a few JCNS that standout in 1971 more than in other years? I still haven`t found why the NOR is higher in 71.

>>PRINT NOR TIMES FOR AIRCRAFT IN 1971 and 1970

>>TOTALS FOR EACH AIRCRAFT< BY YEAR AND SERIAL NUMBER

>>SHOW NOR FOR EACH AIRCRAFT SERIAL NUMBER FOR EACH YEAR STATE AIRCRAFT SERIAL NUMBER< THEN NOR FOR EACH YEAR<STARTING 1970

>>SSHOW PERCENTAGE CHANGE IN NOR FOR EACH AIRCRAFT FOR THE YEAR 1971 list by AIRCRAFT SERIAL NUMBER

>>LIST AIRCRAFT WITH ZERO NOR TIME AND GIVE YEAR

>>DISPLAY PERCENT CHANGE

>>LIST MAINTENANCE FOR AIRCRAFT SERIAL NUMBERS 2,6,8

>>YES PRINT NOR HOURS

>>BREAK DOWN NOR TIME FOR BUSER 2 LIST NORM,NORMS,NORMU,NORS

>>FOR 1971 LIST JCN FOR BUSER 2. LIST JCN AND NOR

>>LIST JCN THAT HAVE A NOR OF GREATER THAN 8 HOURS

>>LIST JCN FOR BUSER2 FOR 1971 THAT HAD REPAIR ACTION T

>>LIST BUSER2 1971 JCN WITH AT CODE T WUC AND ACTWC

>>TOTAL NOR HOURS FOR BUSER2 1971 AT CODE T AND TOTAL NOR HRS FOR 1971

>>LIST AT=T<WUC<NOR FOR BUSER2

>>YES

>>LIST NORS FOR BUSER 6 8 1970,71,72

>>LIST NOR FOR BUSER 2,6,8 FOR 1970,71,72

>>LIST FLIGHT HOURS FOR BUSER 2,6,8 FOR 1970,71,72

>>LISST BUSER6 NOR NORM NORMU NORMS NORS FOR 1970,71,72

>>SAM<E INFO FOR BUSER 2 8

>>Planes data i would like to get down time information on downtime of planes 1971 I would like to get information on the type of work that was done and the type of parts that where either replaced or repaired

>> Planes Data where there any extra parts that where installed on the se aircraftin 1971 that where not installed in 70,72,or73.

>>P Planes Data can you give me the reasons for the down time of the planes in1971 1971, like for example how many where down bbecause of inspection, how many where down because they needed repairs,etc

>>Planes Data give me the hours of scheduled maintenance--unscheduled maintenance and compare it with those for 70, 72, and 73.

>>Planes Data can you give me more information on the average work that was performed on the aircraft during the unscheduled maintenance?

>>Planes Data I don't want to know thw number of hours but I want to know what was done to the planes during the maintenance.

>>Planes Data Iwould like to know if there where any major repairs that would group all the airplanes under one catagory.

>> Planes Data can you give me thAWM times of the planes compared to the AW in 71 compared with the times in 70, 72, and 73.

>>Planes Data was the ACTORG in 71 the same as in 70?

>>Planes Data can I get the number of total flight hours of the planes for 70, 71, 72, and 73.

>>Planes Data think you I think I have found the answer.

>>Planes Data can I get the number of planes that ditched on land or water in 71 as compared with those in 72.

>>Planes Data can I get a number of planes that where under the NORS statis in 71 as compared with those in 72.

>>Planes Data can I get the same information for the years of 70 and 73.

>>was extra equipment installed on all the aircraft in 71 as opposed to any of the other years.

>>how many a/c were down due to nor time in 1971? and of these how many were nors?

>>yes

>>both if possible

>>Can I see the average norm time?

>>Was there a radical change in work centers or orginizations who did the work those two years? 71 and 72?

>>can i see that data again, please?

>>Can you determine if there was a high concentration of one type of HOWMAL and/or WUC for the 2 ACTORGS PE3 and AC2? But first I would like to see the above ACTORG information for 1972.

>>Yes, for 1971 and 1972.

>>Can I get the average NOR,NORS,and NORM data for PE3 and AC2 in 71 or 72?

>>zCan Ifind out the average time the a/c spent on air carriers over the four years?

>>okey doke

>>Is there a difference in pilot techniq

>>uWhat are some of the common down time problems

>>To be more clear, are there any particular systems causing the down time? Example...Brakes and anti-skid system, or hydraulic leaks etc...

>>Has there been an average NORMU time? and if so how much different is it than the average NORMS..

>>Yes if possible

>>Of this NORMU, how much was due to expected damage of hard landings?

>>For each year how many Maintenance actions where related to 311, and 878?

>>Was the average Battle damage (731) higher in 1971 than in the other years? What was the average time aircraft sat waitingawaiting maintenance, in each of the four years?

>>What was the average flight time in each of the four years?

>>Was there any significant airframe and/or power plant modifications done in 1971?

>>Was the high AWM due to a lack of maintenance personal?

>>What was the average time maintenance was held up due to waiting for parts?

>>Was the average maintenance action (731) higher in 1971 than in the other years?

>>Yes, was the average number of maintenance actions due to battle damage (731) higher in 1971 than in the other years?

>>what % of the same aircraft were down in 1971

>>were there certain aircraft that were down more than others

>>qWas there a certain PUC that had more NOR A/C than others

>>Was the maintenance done at PUC 38 and 306 mostly NORS,NORMS or NORMU/

>>Did a certain defect show up more than others//

>>How many cases of FOD were experienced/

>>Consider aircraft 27.

>>How far from a supply base is PUC 38 and 306

>>Is there a record of AWM times for these PUC's

>>Does it have to be obtained using A/C numbers or can I get a complete listing?

>>Consider A/C 46.

>>yes

>>Was the a/c away from it's assigned PUC at the time it became NOR?

>>What was it's FPC.

>>I don't understand these numbers.

>>What WUC was giving the problem?

>>How much NOR time did a/c 34 have in 1971?

>>Was it NORMS,NORMU or NORS that kept this a/c from flying?

>>Is a/c 34 an F4 or A7?

>>What NORMS is considered normal for an A7?

>>Let's use 1971 and1970.

>>What were the total of flight hours recorded in those two years for A7's?

>>Were any A7's retired after '71 and replaced with new a/c?

>>Is it possible to find out how many JCN's are recorded for a/c 34?

>>For JCN 56-Was it NORMS or NORMU?

# VITA

Harry Tennant was born on August 17, 1949 in Chicago, Illinois. He recieved a B.S. degree in Information Engineering and Computer Science from the University of Illinois at Chicago Circle in June, 1973. From 1973 to 1974 he was an electronic engineer in the Quality department of Honeywell, Inc. He worked on building automation systems.

In the fall of 1974 he enrolled in the Graduate College of the University of Illinois at Chicago Circle. He recieved his M.S. degree through the Department of Information Engineering in 1977. His Masters thesis, "The Automatic Advisor," was a natural language question answering system.

In the fall of 1975, he moved to the Urbana-Champaign campus of the University of Illinois where he began studies for his Ph.D. through the Department of Computer Science. He was a research assistant at the Coordinated Science Laboratory.

He is the author of the book, Natural Language Processing: An Emerging Technology and has written a version of LISP for microcomputers.


MED