Out of the Laboratory:
A Case Study with the IRUS Natural Language Interface[1]

by

Ralph M. Weischedel, Edward Walker, Damaris Ayuso, Jos de Bruin,
Kimberle Koile, Lance Ramshaw, Varda Shaked

BBN Laboratories Inc.
10 Moulton St.
Cambridge, MA 02238

Abstract

As part of DARPA's Strategic Computing Program, we have moved a large natural language system out of the laboratory. This involved:

- Delivery of knowledge acquisition software to the Naval Ocean Systems Center (NOSC) to build linguistic knowledge bases, such as dictionary entries and case frames,
- Demonstration of the natural language interface in a naval decision-making setting, and
- Delivery of the interface software to Texas Instruments, which has integrated it into the total software package of the Strategic Computing Fleet Command Center Battle Management Program (FCCBMP).

The resulting natural language interface will be delivered to the Pacific Fleet Command Center in Hawaii.

This paper is an overview of this effort in technology transfer, indicating the technology features that have made this possible and reflecting upon what the experience illustrates regarding transportability, technology status, and delivery of natural language processing outside of a laboratory setting. The paper will be most valuable to those engaged in applying state-of-the-art techniques to deliver natural language interfaces and to those interested in developing the next generation of complete natural language interfaces.

[1] The work presented here was supported under DARPA contract #N00014-85-C-0016. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or of the United States Government.

1 Introduction

DARPA's Strategic Computing Program in the application area of Navy Battle Management has provided us several challenges and opportunities in natural language processing research and development. At the beginning of the effort, a set of domain-independent software components, developed through fundamental research efforts dating back as much as seven years, existed. The IRUS software [1] consists of two subsystems: one for linguistic processing and one for adding specifics of the back end. The first subsystem is linguistic in nature, while the second subsystem is not. Linguistic processing includes morphological, syntactic, semantic, and discourse analysis to generate a formula in logic corresponding to the meaning of an English input. The linguistic subsystem is application-independent and also independent of data base interfaces. (This is achieved by factoring all application specifics into the back end processor or into knowledge bases such as dictionary entries and case frame rules, that are domain-specific.) The non-linguistic components convert the logical form to the code necessary for a given underlying system, such as a relational data base.

The IRUS system, or its components, had been used extensively in the laboratory, not just at BBN, but also in research projects at USC/Information Sciences Institute, the University of Delaware, GTE Research, and General Motors Research. However, it had not been exercised thoroughly outside of a research environment.

Our goals in participating in the Strategic Computing Program are manifold:

- To test the collection of state-of-the-art heuristics for natural language processing with a user community trying to solve their problems on a daily basis.
- To test the heuristics on a broad, extensive domain.
- To incorporate research ideas (which are often developed in relative isolation in the laboratory) into a complete system so that effective evaluation and refinement can occur.
- To continue the feedback loop of incorporating new research ideas, testing them in a complete system with real users, evaluating the results, and refining the research accordingly on a repeated basis for several years.

There are several accomplishments in the first year and a half of this work. First, the IRUS software has been delivered to the Naval Ocean Systems Center (NOSC) so that their team may encode the dictionary information, case frame rules, and transformation rules for generating queries appropriate for the underlying systems. The NOSC staff involves a linguist plus individuals trained in computer science, but does not involve experts in natural language processing nor in artificial intelligence. Second, the natural language interface software has been delivered to Texas Instruments (TI), which has integrated it into the Force Requirements Expert System (FRESH). Demonstrations of the natural language interface are being given at several conferences this year as well as to the navy personnel at the Pacific Fleet Command Center. Testing and evaluation of IRUS, both its software and the knowledge bases defined by NOSC for the FCCBMP, will be carried out in the spring of 1986, by the Navy Personnel Research and Development Center.

In this section and section two we present evidence that this is one of the most ambitious applications and tests of natural language processing ever attempted. Section two provides more background regarding the technical challenges inherent in the application environment and in the goals of the Strategic Computing Program. Section three describes what was changed in each system component to support the technology transfer. Section four presents and illustrates the principles that have been underscored in moving this substantial AI system from the laboratory to use; while some principles may appear like common sense, reporting on all the experience should be valuable to future efforts. Section five briefly discusses possible future directions, while section six states our conclusions.

2 Background Constraints and Goals

The following sections summarize several constraints and goals which have made this not only a demanding challenge for natural language processing but also an ambitious demonstration of the fruit of AI research.

2.1 Multiple Underlying Systems

The decision support environment of the Fleet Command Center Battle Management Program (FCCBMP) involves a suite of decision-making tools. A substantial data base is at the core of those tools and includes roughly 40 relations and 250 fields. In addition, application programs for drawing and displaying maps, various calculations and additional decision support capabilities are provided in the Operations Support Group Prototype (OSGP). In a parallel part of the Strategic Computing Program, two expert systems are being provided: the Force Requirements Expert System (FRESH) and the Capabilities Assessment Expert System (CASES). TI is building the FRESH expert system; the contract for the CASES expert system has not been awarded as of the writing of this paper.

The target users are navy commanders involved in decision making at the Pacific

Fleet Command Center; these are top-level executives whose energy is best spent on navy problems and decision making rather than on the details of which of four underlying systems offers a given information capability, on how to divide a problem into the various information capabilities required and how to synthesize the results into the desired answer. Currently they do not access the data base or OSGP application programs themselves; rather, on a round-the-clock basis, two operators are available as intermediates between commander and computer. Consequently, the need for a natural language interface (NLI) is paramount.

2.2 The Need For Transportability

There are three ways that transportability has been absolutely required for the natural language interface. First, since we had no experience previously with this application domain, and since the schedule for demonstrations and delivery was highly ambitious, only the application-independent software could be brought to bear on the problem initially; therefore, transportability across application domains was required. Second, the underlying systems have been and will continue to be evolving. For instance, the data base structure is being modified both to support additional information needs for the new expert systems and to provide shorter response time in service of human requests and expert system requests to the data base.

Third, the target output of the natural language interface is subject to change. For instance, the capabilities of FRESH are being developed in parallel with the natural language interface and the CASES expert system has not been started as of this date. Interestingly enough, the target language for the data base could change as well. For instance, there is the possibility of replacing the ORACLE data base management system with a data base machine, in which case the target language would change though the application and data base structure remained constant during the period of installing the data base machine.

2.3 Technology Testbed

The project has two goals which at first seem to conflict. First, the software must be hardened enough to be an aid in the daily operations of the Fleet Command Center. Second, the delivered systems are to be a testbed for research results; feedback from use of the systems is to provide a solid empirical base for suggesting new areas of research and refinement of existing research.

As a consequence, software engineering demands placed upon the AI software are quite rigorous. The architecture of the software must support high quality, well worked out, non-toy systems. The software must also support substantial evolution in the heuristics and methods employed as natural language processing provides new research ideas that can be incorporated.

3 Adequacy of the Components

In this section we present a brief analysis of the adequacy of the various components in the system, given that the software had not been built with this domain in mind (but had been built with transportability in mind) and given that one of the goals of the effort is to provide a flexible technological base allowing evolution of the techniques and heuristics employed.

3.1 Knowledge Representation

At the start of the project, the underlying knowledge representation consisted of a hierarchy of concepts (unary predicates), a list of functions on instances of those concepts, and a list of n-ary predicates. The knowledge representation served several purposes:

- To identify the predicate symbols and function symbols that could be used in the first order logic representing the meaning of sentences,
- To validate selection restrictions (case frame constraints) during the parsing process.

Early on we concluded that greater inference capabilities were required. We wanted to be able to:

- State and reason about knowledge of binary relationships. For instance, every vessel has an arbitrary number of overall readiness ratings associated with it, corresponding to the history of its readiness.
- Represent events and states of affairs flexibly. There may be a variable number of arguments expressed in the input for a given event. For instance, *Admiral Foley deployed the Eisenhower yesterday* or *Admiral Foley deployed the Eisenhower C3.*[2] Also, we needed to be able to count occurrences of events or states of affairs over history, as in *How many times was the the Eisenhower C3 in the last 12 months?* Consequently, we have chosen to represent events and states of affairs as entities, which participate in a number of binary relationships, for instance, specifying the agent, time, location, etc. of the event.

Therefore, the initial ad hoc knowledge representation formalism was replaced with a more general framework, NIKL [10], the new implementation of KL-ONE. This met the needs stated above, and also provided inference mechanisms [15] which could serve as

[2] *C3* is an overall readiness rating.

a partial consistency checker on the axioms for the navy domain. Of course, there are other ways to achieve the goals above. However, NIKL was available, and this would be its first use in a technology transfer effort, providing us the opportunity to further explore the power and limitations of limited inference systems.

In NIKL, one can state the classes of entities, the binary relations between entities (including functional relationships), subclass relationships, and subsumption relations among binary relations. It is now used to support:

- The validation of selection restrictions during the parsing process,
- Proposal of possible case frame constraints and possible predicates by the semantic knowledge acquisition component,
- Proposal of the meaning of vague relationships, such as "have", and
- The mapping from first-order logic to relational data base queries.

Once the more powerful knowledge representation and inference mechanisms [15] were available to IRUS, we began using them in unanticipated ways, for instance, the last three in the list above.

3.2 The Lexicon and Grammar

The current grammar (RUS) [2] and lexicon are based on the ATN formalism [23]. Though RUS was designed to be a general grammar of dialogue and was clearly among a handful of implemented grammars having the broadest coverage of English, the question was how much modification would be needed for the Navy domain, which was totally new to us.

Very few changes were needed to the software that supports the lexicon and morphological analysis. Those that were required centered around special military forms, such as allowing *06Mar86* as a date and *0600z* as a time. Special symbols and codes such as those are bound to arise in many applications, no matter how transportable the software is.

Very few modifications to the grammar had to be made; those that have been made thus far correspond to special forms and have required very little effort to add. Examples include military (and European) versions of dates, such as *6 March 1986*. This is not to claim that everything a navy user types will be parsed; fully general treatments for conjunction, gapping, and ellipsis, are still research issues for us, as for everyone else. Rather, the experience testifies to the fact that domain-independent grammars can be written for natural language interfaces and that modification of them for a new application can be very small. Sager [12] has reported that few rules of the Linguistic String Parser need to be changed when it is moved to a new application.

The current system handles several classes of ill-formed input, including typographical errors that result in an unknown word; omitted words such as determiners and prepositions; various grammatical errors such as subject verb disagreement and determiner noun disagreement; case errors in using pronouns; and elliptical inputs. The strategy is that of [21].

3.3 Semantic Interpretation

Though the software for the semantic interpreter did not depend on domain specifics, the limitations of the initial knowledge representation formalism and of the class of linguistic expressions for which it could compute a semantic representation meant that the semantic interpreter had to be substantially changed. First, the semantic interpreter was modified to take advantage of the stronger knowledge representation formalism and inference available in NIKL. For instance, the interpreter must compute the semantic representation for descriptions of events and states of affairs. It now finds the interpretation of *X has Y* by looking for a relation in the knowledge representation between X and Y.

Second, the semantic interpreter has been changed to correspond more and more to general linguistic analysis. One strength of the initial version of the semantic interpreter [1] was its ability to handle idiomatic expressions, such as *blue forces*. *Blue forces* refers to U.S. forces, as opposed to forces that are blue (in color). The semantic interpreter has been generalized now so that it is much easier to capture the general meaning of *blue* as a predicate, as well as allowing for specification of idiomatic expressions, such as *blue forces*.

A major focus in the next year will be continuing modification of the semantic interpreter so that we have a fully compositional semantics and an intensional logic, rather than a first order logic as the meaning representation of a given sentence. The compositional semantics will still allow, of course, for idiomatic expressions. The enhanced semantic interpreter will be applicable to a much broader class of English expressions, while still being domain-independent and driven by domain-specific case frame rules.

The semantic interpreter does not allow for semantic ill-formedness at present; removing this restriction is a high priority research area.

3.4 Discourse Phenomena

Since discourse analysis is the least understood area in natural language processing, the discourse processing component in the system is limited. The system handles anaphora based on the class of the entity required by the selection restrictions upon the anaphor. A benefit of the change in representation making events and states of affairs entities is that the simple heuristic above allows the anaphor in each of the following sequences to be correctly understood:

- *The Eisenhower was deployed C2. When did that occur?*
- *The Eisenhower had been C3. When was that?*

Elliptical inputs that are noun phrases or prepositional phrases are handled as follows: If the class of the entity inherent in the elliptical input is consistent with a class in the previous input, the semantic representation of the new entity is substituted for the semantic representation in the previous input. If not, the ellipsis is interpreted as a request to display the appropriate information.

Far more sophisticated discourse processing is a high priority not only for our project but for natural language work altogether.

3.5 Introducing Back end Specifics

The result of linguistic processing in IRUS is a formula in logic. Another component translates the logical expression representing the meaning of an input into an expression in an abstract relational algebra. Simple optimization of the resulting expression is performed in the same component. The initial version of that component (MRLtoERL) [17] used local transformations to translate the n-ary predicates of the logic into the appropriate sequence of projections, joins, etc. on files and fields of the data base.

A straightforward, syntax-directed code generator translates the abstract relational expression into the query language required by the underlying data base management system. Code generators have been built for System 1022, the Britton-Lee Data Base Machine, and ORACLE. An experienced person needs only two to three weeks to create the code generator.

With the move to NIKL and the representation of events and states of affairs as concepts participating in binary relations, the context-free translation of predicates to expressions in relational algebra was no longer adequate. However, the limited inference mechanism [15] of NIKL formed a basis for a simplifier [18] as a preprocess to the MRLtoERL component so that the translation from logic to relational algebra could still be done using only local transformations. Furthermore, the simplifier enabled general translation of linguistic expressions whose data base structure bears little resemblance to the conceptual structure of the English query [18]. We believe the simplification techniques can be generalized further to support the simplification of a subclass of expressions in the intensional logic to be generated by the planned semantic interpreter [19].

Introduction of back end specifics for the OSGP application package and the FRESH expert system is handled by an ad hoc translator from logic to target code at present.

3.6 Linguistic Knowledge Acquisition

IRUS's four knowledge bases are:

- The lexicon, which states syntactic and morphological information,
- The taxonomy of case frame rules,
- The model of predicates in the domain, stated in NIKL, and
- The transformation rules for mapping predicates in the logic into projections, joins, etc. of fields in the data base.

The first two of these are linguistic knowledge bases; sophisticated acquisition tools are available to aid the system builder, though not necessarily trained in AI, to build the necessary linguistic knowledge about the vocabulary.

Powerful knowledge acquisition tools for building these domain-specific constraints could greatly ease the process of bringing up a natural language interface for a new application and consequently for broadening the applicability of NLI technology. Perhaps the most powerful demonstration of acquisition tools to date has been TEAM [6]. Based on the fields and files of a given data base, TEAM's acquisition tools lead the individual through a sequence of questions to acquire the specific linguistic and domain knowledge needed to understand a broad subset of language for querying the data base. However, since those heuristics are in large part specific to the task of accessing data bases, that technology could not be directly applied to the FCCBMP application, which encompasses a relational data base, an application package including both map drawing and calculation, and expert systems.

Knowledge acquisition tools for IRUS, developed under earlier DARPA-funded work at BBN, were not specific to data base applications and therefore could be applied in the FCCBMP. Even if applicability of the TEAM heuristics were not a problem, there are theoretical and technical difficulties in translating English requests into data base queries [9] which would argue for a more general approach such as ours. As Scha [13, 14] has argued, these difficulties, as well as the issues of transportability and generality, suggest keeping linguistic knowledge rather independent of assumptions about the back end.

IRACQ, the semantic acquisition tool made available to NOSC for specifying case frames and their associated translations, is quite powerful. The initial version [11] allowed one to specify the case frame for a new word sense by giving an example of a phrase using that word sense. For instance, if *the admiral, a vessel,* and *C2* are known to the system, then one can define a new case frame for *deploy* by giving a phrase such as *the admiral deployed a vessel C2.* The system suggests generalizations of the arguments specified in the example using the NIKL knowledge base, so that the inferred case frame is the most general that the user authorizes. For example, generalizations of admiral are commanding officer, person, and physical object; generalizations of vessel are unit, platform, and physical object; generalizations of C2 are rating and code. Furthermore, based on the introduction of the more general knowledge representation system NIKL, IRACQ is being extended to propose the binary relations that might be part of the translation of the new word. Of course, if the relations and concepts needed are not already present in the domain predicate model, the user can define new concepts and relations in the NIKL hierarchy as well.

The availability of such knowledge acquisition tools has made it possible for NOSC representatives, rather than AI experts, to define the naval language expected as input. We have found that even with the tool described above, reasonable linguistic sophistication is very helpful in defining the case frames. In fact, an individual with a master's degree in linguistics is defining the case frames at NOSC. More sophisticated tools, which do not presuppose only one kind of back end, are one of the most important research topics for natural language interfaces. These would combine the strengths of the linguistic knowledge acquisition tools of both IRUS and TEAM.

4 Principles Underscored

In the course of the effort, a number of principles have been underscored. Many of these once stated may appear to be common sense; however, we hope that illustrating them from our experience will prove helpful.

4.1 The Necessity For General Solutions

The availability of domain-independent software driven by domain-dependent, declarative knowledge bases was of paramount importance because of the following:

- The application was not only broad (three underlying systems) but also evolving (with a fourth system to be added).
- Great habitability is necessary for delivery to the Pacific Fleet Command Center.
- The time frame for demonstration was relatively short compared to the scope of the underlying systems to be covered.

Furthermore, it is critical that the knowledge bases state a linguistic or domain fact once and that the domain-independent software be able to use that one fact in all predictable linguistic variations. The reasons are obvious: the efficiency in building the knowledge bases, the consistency of stating a fact only once, and the habitability of the resulting system which can understand things no matter what form they are expressed in.[3]

The IRUS system attains the goal mentioned above relatively well; a linguistic or application constraint is stated once in the knowledge base but applied in all possible ways in the language processing. This is particularly true because of the substantial grammar [2, 3] and to a lesser extent due to the semantic interpreter. Recognition of this fact is part of the reason that substantial changes, as mentioned in section three, are planned in the semantic interpreter to make the linguistic facts that drive it even more general.

[3] An interesting anecdote that arose in early discussions in the planning of this project centered around the tight deadlines and the breadth of the application area. Since it was clear that one could not cover all three underlying systems in every area for which they could provide information, the question arose whether to focus on a substantial subpart of the application domain initially or to sacrifice linguistic coverage to gain in coverage of the underlying systems. Because the information needs of the various navy personnel differed widely, and because the scope of needs seemed impossible to predict, navy personnel initially suggested that coverage of all possible information stored in the underlying systems was of such importance that sacrifices regarding the language understood could be made even if there were only one way that a given piece of information could be accessed. The interesting thing however is that as demonstrations were given, the first things people request following the demonstration is to try various rephrasings of the requests in the demonstration, thereby in behavior indicating how important not being restricted to special forms is.

4.2 The Necessity of Heuristic Solutions

In the previous section we have argued for the need of general purpose solutions to problems in NLI. Clearly this cannot be taken to an extreme; otherwise one would not have an NLI in the foreseeable future, since there are well-known outstanding problems for which there is no general, comprehensive solution on the horizon. Consequently, heuristic, state-of-the-art solutions are being demonstrated for problems such as ambiguity, vagueness, discourse context, ill-formed input, definite reference, quantifier scope, conjunction, and ellipsis. Though laboratory use of the system embodying that set of heuristics is quite promising, we expect that placing the system in the hands of individuals trying to solve their day-to-day problems will produce interesting corpora of dialogues that cannot be handled by one or more of those heuristics. Careful study of those corpora will tell us not only the effectiveness of state-of-the-art solutions but will also suggest new directions of research.

4.3 The Necessity of Extra-linguistic Elements in a Natural Language Interface

Having only a natural language processor is not sufficient to provide a truly natural interface. Four elements seem highly valuable for typed input: editing, a readily accessible history of the session, human factors elements in the presentation, and a minimum of key strokes. Editing should include more than deleting the last character of the string and deleting the whole string. We are currently relying on Emacs, which is readily available on Symbolics workstations. However, that is also unattractive because of the arcane nature of the link between the myriad control key commands of Emacs and the actual textual tasks the user needs to perform.

IRUS's on-line history of the session provides reviewing earlier results, editing the text of earlier requests to create new ones, and generating a standard protocol for routine operations that occur on a regular basis. Our user community anticipates a need for both routine sequences of questions as would be useful in preparing daily or weekly reports, and ad hoc queries, e.g., when crises arise.

Issues in presentation are important as well. No matter what the underlying application is, IRUS lets it produce output on the complete bitmap screen. A popup input window and an optional popup history window can be moved to any part of the screen so that all parts of the underlying system's output may be visible.

Certain operations occur so frequently that one would like to have them available on the screen at all times in menus to minimize memory load and key strokes. Examples are clearing a window and aborting a request.

A future capability that would be quite attractive is pointing to individual data items, classes of data items, field headings, or locations on maps, causing the appropriate linguistic description of that entity to be made available as part of the natural language input. While this is possible in the future, providing such a capability is not currently funded.

Speech input as a mode of communication would also be highly desirable, even if extremely limited initially. As a consequence, the next generation of natural language understanding systems in the FCCBMP will include modifications specifically to provide an infrastructure which could at a later date support speech input.

5 Future Possibilities

In addition to the enhancements we have mentioned earlier regarding the semantic interpreter, linguistic knowledge acquisition tools, and discourse processing, there are three substantial areas of research and development possible. First, research in ill-formed input is necessary in order to allow for additional grammatical problems in the input and for relaxation of semantic constraints, e.g., to allow for figures of speech. The problem with an ill-formed input is that there is no interpretation which satisfies all linguistic constraints. Therefore, the very constraints that limit search must be relaxed, thereby opening Pandora's Box in terms of the number of alternatives in the search space. Not only IRUS, but apparently all systems that process any ill-formed input attain the success they do by considering very few kinds of ill-formed input and by assuming that semantic constraints can never be violated.[4] Consequently, determining what the user meant in an ill-formed input is a substantial problem requiring research.

Second, we propose exploring parallel architectures to add functional capability. Run time performance of IRUS on a Symbolics machine is quite acceptable. Typical inputs are fully processed to give the target language input to the underlying system within a few seconds; naturally, the relational data base and underlying expert systems are not expected to be able to perform at comparable speeds. There are three areas where functional performance could be improved by parallelism.

1. The current system ranks the partial parses using both semantic and syntactic information, and it explores those partial parses based on following up the most promising one first. The technique is relatively effective, but clearly not infallible. Finding all interpretations and then

[4] Early work on allowing semantic relaxation is reported in [5, 21, 22].

ranking them based not only on local syntactic and semantic tests but also on global semantic,. pragmatic, and discourse information is critical to improving the identification of what the user intended.

2. A second area related to the first, is greater coverage of ill-formed input. As mentioned earlier, ill-formedness requires relaxing the rules that constrain search; therefore the search space grows dramatically in processing an ill-formed input.

3. Real-time, large vocabulary, large branching factor, continuous speech recognition is beyond the state of the art, and requires highly parallel machines to support speech signal processing. While this is highly desirable, it is not part of our current effort.

Within the next two years we intend to replace the ATN grammar with a declarative, side-effect free grammar and a parallel parsing algorithm, following work reported in [16].

Third, our evolving system is being interfaced to the Penman generation component from USC/Information Sciences Institute (USC/ISI) [8]. Penman is based upon systemic linguistics. The ultimate goal of the effort with USC/ISI is twofold: to have systems that can understand whatever they generate and to achieve this by having common knowledge sources for the lexicon, for the NIKL model of domain predicates, and for discourse information.

6 Conclusions

Though the project will be ongoing for several years yet, there are several preliminary conclusions from the first year and a half of effort, given the constraints and goals mentioned in section two.

1. Providing language coverage for this broad application with multiple underlying systems has not been a problem. However, since determining what system(s) must be accessed for a given input is a research problem that has been little addressed, only simple linguistic clues are used in the current version. The problem in general involves not only reasoning about the capabilities of the underlying systems [7] but also significant linguistic issues. For instance, if one says *Show me the carriers whose condition code changed in the last 24 hours*, either a list (from the data base) or a map (from OSGP) is appropriate. If one says *Show me a display of the carriers whose condition code changed in the last 24 hours*, only OSGP is appropriate. The linguistic cue is *display*. Furthermore, some contexts favor one underlying system over the other, requiring the system to maintain a dialogue context model, including the user's inferred goals in the dialogue, in order to integrate cues from dialogue context with the linguistic cues.

2. The architecture has supported transportability well. For instance, this new application required only minor changes to the grammar and morphological analyzer. As FRESH has been further defined and as the data base structure has evolved, only small local changes have been required to the content of the knowledge bases. Should a data base machine replace the

current data base management system in Hawaii, only two to three person weeks should be needed to generate the new target language. However, more sophisticated linguistic knowledge acquisition tools not dependent on the type of the underlying application system are a critical goal for NLI both for far greater applicability of the technology and for far broader availability of NLIs.

3. The success of this effort as a technology testbed depends on evaluation after installation at the Pacific Fleet Command Center and on the success of the architecture to support substantial enhancements, such as the planned semantic interpreter based on compositional semantics and the planned parallel parser. However, it already has supported massive changes well, such as the change in underlying knowledge representation when NIKL was introduced.

The potential of the testbed is great because it offers empirical research of a realistic kind unfortunately largely lacking heretofore; the placement of TQA in the hands of users to solve their daily problems for a year [4] is a notable exception. The results of research on heuristics for definite reference; semantic ambiguity; ellipsis; syntactically or semantically ill-formed input; and inference from world knowledge and context, to name a few studied in isolation, must be tested in a complete system. The opportunity in the FCCBMP will help to determine the effectiveness of such heuristics in a large diverse application domain where combinatorial issues cannot be ignored. Collecting corpora in an experiment can be highly instructive, as shown in [20]. However, corpus collection using people solving their own problems provides an uncommon degree of realism and legitimacy to the empirical process.

References

[1] Bates, M., Stallard, D., and Moser, M.
The IRUS Transportable Natural Language Database Interface.
Expert Database Systems.
Cummings Publishing Company, Menlo Park, CA, 1985.

[2] Bobrow, R.J.
The RUS System.
In B.L. Webber, R. Bobrow (editors), *Research in Natural Language Understanding.*Bolt, Beranek and Newman, Inc., Cambridge, MA, 1978.
BBN Technical Report 3878.

[3] Bobrow, R. and Bates, M.
The RUS Parser Control Structure.
In *Research in Knowledge Representation for Natural Language Understanding, Annual Report.*Bolt Beranek and Newman Inc., 1982.
BBN Report No. 5188.

[4] Damerau, F.J.
Operating Statistics for the Transformational Question Answering System.
American Journal of Computational Linguistics 7(1):30–42, 1981.

[5] Fass, D. and Wilks, Y.
Preference Semantics, Ill–Formedness, and Metaphor.
American Journal of Computational Linguistics 9(3–4):178–187, 1983.

[6] Grosz, B., Appelt, D. E., Martin, P., and Pereira, F.
TEAM: An Experiment in the Design of Transportable Natural Language Interfaces.
Technical Report 356, SRI International, 1985.
To appear in Artificial Intelligence.

[7] Kaczmarek, T., Mark, W., and Sondheimer, N.
The Consul/CUE Interface: An Integrated Interactive Environment.
In *Proceedings of CHI '83 Human Factors in Computing Systems*, pages 98–102. ACM, December, 1983.

[8] Mann, W.C. and Matthiessen, C.M.I.M.
Nigel: A Systemic Grammar for Text Generation.
Systemic Perspectives on Discourses: Selected Theoretical Papers from the 9th International Systemic Workshop.
Ablex, Norwood, NJ, forthcoming.

[9] Moore, R.C.
Natural Language Access to Databases – Theoretical/Technical Issues.
In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 44–45. Association for Computational Linguistics, June, 1982.

[10] Moser, M.G.
An Overview of NIKL, the New Implementation of KL–ONE.
In Sidner, C. L., et al. (editors), *Research in Knowledge Representation for Natural Language Understanding – Annual Report, 1 September 1982 – 31 August 1983*, pages 7–26.BBN Laboratories Report No. 5421, 1983.

[11] Moser, M.G.
Domain Dependent Semantic Acquisition.
In *The First Conference on Artificial Intelligence Applications*, pages 13–18. IEEE Computer Society, December, 1984.

[12] Sager, N.
The String Parser for Scientific Literature.
In R. Rustin (editor), *Natural Language Processing*, pages 61–88.Algorithmics Press, Inc., New York, NY, 1973.

[13] Scha, R.J.H.
English Words and Data Bases: How to Bridge the Gap.
In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 57–59. Association for Computational Linguistics, June, 1982.

[14] Scha, R.J.H.
Logical Foundations for Question Answering.
Technical Report, Eindhoven: Philips Research Labs, M.S. 12.331., 1983.

[15] Schmolze, J.G., Lipkis, T.A.
Classification in the KL–ONE Knowledge Representation System.
In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence.* 1983.

[16] Sridharan, N.S.
Semi–Applicative Programming: Examples of Context Free Recognizers.
Technical Report Report No. 6135, BBN Laboratories Inc., January, 1986.

[17] Stallard, D.
Data Modelling for Natural Language Access.
In *The First Conference on Artificial Intelligence Applications*, pages 19–24. IEEE Computer Society, December, 1984.

[18] Stallard, D.G.
A Terminological Simplification Transformation for Natural Language Question–Answering Systems.
In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics, July, 1986.
To appear.

[19] Stallard, D.G.
Taxonomic Inference on Predicate Calculus Expressions.
Technical Report, BBN Laboratories Inc., 1986.
In preparation.

[20] Thompson, B.H.
Linguistic Analysis of Natural Language Communication with Computers.
In *Proceedings of the Eighth International Conference on Computational Linguistics*, pages 190–201. International Committee on Computational Linguistics, October, 1980.

[21] Weischedel, R. M. and Sondheimer, N. K.
Meta–rules as a Basis for Processing Ill–Formed Input.
American Journal of Computational Linguistics 9(3–4):161–177, 1983.

[22] Weischedel, R.M. and Sondheimer, N.K.
Relaxing Constraints in MIFIKL.
Technical Report, USC/Information Sciences Institute, 1983.

[23] Woods, W.A.
Transition Network Grammars for Natural Language Analysis.
CACM 13(10):591–606, October, 1970.