AN ENVIRONMENT FOR ACQUIRING SEMANTIC INFORMATION

Damaris M. Ayuso, Varda Shaked, and Ralph M. Weischedel
BBN Laboratories Inc.
10 Moulton St.
Cambridge, MA 02238

Abstract

An improved version of IRACQ (for Interpretation Rule ACQuisition) is presented.[1] Our approach to semantic knowledge acquisition: 1) is in the context of a general purpose NL interface rather than one that accesses only databases, 2) employs a knowledge representation formalism with limited inferencing capabilities, 3) assumes a trained person but not an AI expert, and 4) provides a complete environment for not only acquiring semantic knowledge, but also maintaining and editing it in a consistent knowledge base. IRACQ is currently in use at the Naval Ocean Systems Center.

1 Introduction

The existence of commercial natural language interfaces (NLI's), such as INTELLECT from Artificial Intelligence Corporation and Q&A from Symantec, shows that NLI technology provides utility as an interface to computer systems. The success of all NLI technology is predicated upon the availability of substantial knowledge bases containing information about the syntax and semantics of words, phrases, and idioms, as well as knowledge of the domain and of discourse context. A number of systems demonstrate a high degree of transportability, in the sense that software modules do not have to be changed when moving the technology to a new domain area; only the declarative, domain specific knowledge need be changed. However, creating the knowledge bases requires substantial effort, and therefore substantial cost. It is this assessment of the state of the art that causes us to conclude that *knowledge acquisition is one of the most fundamental problems to widespread applicability of NLI technology.*

This paper describes our contribution to the acquisition of semantic knowledge as evidenced in IRACQ (for Interpretation Rule ACQuisition), within the context of our overall approach to representation of domain knowledge and its use in the IRUS natural language system [5, 6, 27]. An initial version of IRACQ was reported in [19]. Using IRACQ, mappings between valid English constructs and predicates of the domain may be defined by entering sample phrases. The mappings, or interpretation rules (IRules), may be defined for nouns, verbs, adjectives, and prepositions. IRules are used by the semantic interpreter in enforcing selectional restrictions and producing a logical form as the meaning representation of the input sentence.

IRACQ makes extensive use of information present in a model of the domain, which is represented using NIKL [18, 21], the terminological reasoning component of KL-TWO [26]. Information from the domain model is used in guiding the IRACQ/user interaction, assuring that acquisition and editing yield IRules consistent with the model. Further support exists for the IRule developer through a flexible editing and debugging environment. IRACQ has been in use by non-AI experts at the Naval Ocean Systems Center for the expansion of the database of semantic rules in use by IRUS.

This paper first surveys the kinds of domain specific knowledge necessary for an NLI as well as approaches to their acquisition (section 2). Section 3 discusses dimensions in the design of a semantic acquisition facility, describing our approach. In section 4 we describe IRules and how they are used. An example of a clause IRule definition using IRACQ is presented. Section 5 describes initial work on an IRule paraphraser. Conclusions are in section 6.

2 Kinds of Knowledge

One kind of knowledge that must be acquired is lexical information. This includes morphological information, syntactic categories, complement structure (if any), and pointers to semantic information associated with individual words. Acquiring lexical information may proceed by prompting a user, as in TEAM [13], IRUS [7], and JANUS [9]. Alternatively, efforts are underway to acquire the information directly from on-line dictionaries [3, 16].

Semantic knowledge includes at least two kinds of information: selectional restrictions or case frame constraints which can serve as a filter on what makes sense semantically, and rules for translating the word senses present in an input into an underlying semantic representation. Acquiring such selectional restriction information has been studied in TEAM, the Linguistic String Parser [12], and our system. Acquiring the meaning of the word senses has been studied by several individuals, including [11, 17]. This paper

[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.

focuses on acquiring such semantic knowledge using IRACQ.

Basic facts about the domain must be acquired as well. This includes at least taxonomic information about the semantic categories in the domain and binary relationships holding between semantic categories. For instance, in the domain of Navy decision-making at a US Fleet Command Center, such basic domain facts include:

All submarines are vessels.
All vessels are units.
All units are organizational entities.
All vessels have a major weapon system.
All units have an overall combat readiness rating.

Such information, though not linguistic in nature, is clearly necessary to understand natural language, since, for instance, "Enterprise's overall rating" presumes that there is such a readiness rating, which can be verified in the axioms mentioned above about the domain. However, this is clearly not a class of knowledge peculiar to language comprehension or generation, but is in fact essential in any intelligent system. General tools for acquiring such knowledge are emerging; we are employing KREME [1] for acquiring and maintaining the domain knowledge.

Knowledge that relates the predicates in the domain to their representation and access in the underlying systems is certainly necessary. For instance, we may have the unary predicates *vessel* and *harpoon.capable*; nevertheless, the concept (i.e., unary predicate) corresponding to the logical expression *(λ x) [vessel(x) & harpoon.capable(x)]* may correspond to the existence of a "y" in the "harp" field of the "uchar" relation of a data base. TEAM allows for acquisition of this mapping by building predicates "bottom-up" starting from database fields. We know of no general acquisition approach that will work with different kinds of underlying systems (not just databases). However, maintaining a distinction between the concepts of the domain, as the user would think of those concepts, separate from the organization of the database structure or of some other underlying system, is a key characteristic of the design and transportability of IRUS.

Finally, a fifth kind of knowledge is a set of domain plans. Though no extensive set of such plans has been developed yet, there is growing agreement that such a library of plans is critical for understanding narrative [20], a user's needs [22], ellipsis [8, 2], and ill-formed input [28], as well as for following the structure of discourse [14, 15]. Tools for acquiring a large collection of domain plans from a domain expert, rather than an AI expert, have not yet appeared. However, inferring plans from textual examples is under way [17].

3 Dimensions of Acquiring Semantic Knowledge

We discuss in this section several dimensions available in designing a tool for acquiring semantic knowledge within the overall context of an NLI. In presenting a partial description of the space of possible semantic acquisition tools, we describe where our work and the work of several other significant, recently reported systems fall in that space of possibilities.

3.1 Class of underlying systems.

One could design tools for a specific subclass of underlying systems, such as database management systems, as in TEAM [13] and TELI [4]. The special nature of the class of underlying systems may allow for a more tailored acquisition environment, by having special-purpose, stereotypical sequences of questions for the user, and more powerful special-purpose inferences. For example, in order to acquire the variety of lexical items that can refer to a symbolic field in a database (such as one stating whether a mountain is a volcano), TEAM asks a series of questions, such as "Adjectives referencing the positive value?" (e.g., *volcanic*), and "Abstract nouns referencing the positive value?" (e.g., *volcano*). The fact that the field is binary allows for few and specific questions to be asked.

The design of IRACQ is intended to be general purpose so that any underlying system, whether a data base, an expert system, a planning system, etc., is a possibility for the NLI. This is achieved by having a level of representation for the concepts, actions, and capabilities of the domain, the *domain model*, separate from the model of the entities in the underlying system. The meaning representation for an input, a logical form, is given in terms of predicates which correspond to domain model concepts and roles (and are hence referred to as *domain model predicates*). IRules define the mappings from English to these domain model predicates. In our NLI, a separate component then translates from the meaning representation to the specific representation of the underlying system [24, 25]. IRACQ has been used to acquire semantic knowledge for access to both a relational database management system and an ad hoc application system for drawing maps, providing calculations, and preparing summaries; both systems may be accessed from the NLI without the user being particularly aware that there are two systems rather than one underneath the NLI.

3.2 Meaning representation.

Another dimension in the design of a semantic knowledge acquisition tool is the style of the underlying semantic representation for natural language input. One could postulate a unique predicate for almost every word sense of the language. TEAM

seems to represent this approach. At some later level of processing than the initial semantic acquisition, a level of inference or question/answering must be provided so that the commonalities of very similar word senses are captured and appropriate inferences made. A second approach seems to be represented in TELI, where the meaning of a word sense is translated into a boolean composition of more primitive predicates. IRACQ represents a related approach, but we allow a many-to-one mapping between word senses and predicates of the domain, and use a more constraining representation for the meaning of word senses. Following the analysis of Davidson [10] we represent the meaning of events (and also of states of affairs) as a conjunction of a single unary predicate and arbitrarily many binary predicates. Objects are represented by unary predicates and are related through binary relations. Using such a representation limits the kind and numbers of questions that have to be asked of the user by the semantic acquisition component. The representation dovetails well with using NIKL [18, 21], a taxonomic knowledge representation system with a formal semantics, for stating axioms about the domain.

3.3 Model of the domain

One may choose to have an explicit, separate representation for concepts of the domain, along with axioms relating them. Both IRUS and TEAM have explicit models. Such a representation may be useful to several components of a system needing to do some reasoning about the domain. The availability of such information is a dimension in the design of semantic acquisition systems, since domain knowledge can streamline the acquisition process. For example, knowing what relations are *allowable* between concepts in the domain, aids in determing what predicates can hold between concepts mentioned in an English expression, and therefore, what are valid semantic mappings (IRules, in our case).

Our NIKL representation of the domain knowledge, the domain model, forms the semantic backbone of our system. Meaning is represented in terms of domain model predicates; its hierarchy is used for enforcing selectional restrictions and for IRule inheritance; and some limited inferencing is done based on the model. After semantic interpretation is complete, the NIKL classification algorithm is used in simplifying and transforming high level meaning expressions to obtain the underlying systems' commands [25]. Due to its importance, the domain model is developed carefully in consultation with domain experts, using tools to assure its correctness. This approach of developing a domain model independently of linguistic considerations or of the type of underlying system is to be distinguished from other approaches where the domain knowledge is shaped mostly as a side effect of other processes such as lexical acquisition or database field specification.

3.4 Assumptions about the user of the acquisition tool.

If one assumes a human in the semantic acquisition process, as opposed to an automatic approach, then expectations regarding the training and background of that user are yet another dimension in the space of possible designs. The acquisition component of TELI is designed for users with minimal training. In TEAM, database administrators or those capable of designing and structuring their own database use the acquisition tools. Our approach has been to assume that the user of the acquisition tool is sophisticated enough to be a member of the support staff of the underlying system(s) involved, and is familiar with the way the domain is conceived by the end users of the NLI. More particularly, we assume that the individual can become comfortable with logic so that he/she may recognize the correctness of logical expressions output by the semantic interpreter, but need not be trained in AI techniques. A total environment is provided for that class of user so that the necessary knowledge may be acquired, maintained, and updated over the life cycle of the NLI. We have trained such a class of users at the Naval Ocean Systems Center (NOSC) who have been using the acquisition tools for approximately a year and a half.

3.5 Scope of utilities provided.

It would appear that most acquisition systems have focused on the inference problem of acquiring knowledge initially and have paid relatively little attention to explaining to the user what knowledge has been acquired, providing sophisticated editing facilities above the level of the internal data structures themselves, or providing consistency checks on the database of knowledge acquired. Providing such a complete facility is a goal of our effort; feedback from non-AI staff using the tool has already yielded significant direction along those lines. The tool currently has a very sophisticated, flexible debugging environment for testing the semantic knowledge acquired independently of the other components of the NLI, can present the knowledge acquired in tables, and uses the set of domain facts as a way of checking the consistency of what the user has proposed and suggesting alternatives that are consistent with what the system already knows. Work is also underway on an intelligent editing tool guaranteeing consistency with the model when editing, and on an English paraphraser to express the content of a semantic rule.

4 IRACQ

The original version of IRACQ was conceived by R. Bobrow and developed by M. Moser [19]. From sample noun phrases or clauses supplied by the user, it inferred possible selectional restrictions and let the user choose the correct one. The user then had to supply the predicates that should be used in the interpretation of the sample phrase, for inclusion in the IRule.

From that original foundation, as IRUS evolved to use NIKL, IRACQ was modified to take advantage of the NIKL knowledge representation language and the form we have adopted for representing events and states of affairs. For example, now IRACQ is able to suggest to the user the predicates to be used in the interpretation, assuring consistency with the model. Following a more compositional approach, IRules can now be defined for prepositional phrases and adjectives that have a meaning of their own, as opposed to just appearing in noun IRules as modifiers of the head noun. Thus possible modifiers of a head noun (or nominal semantic class) include its complements (if any), and only prepositional phrases or other modifiers that do not have an independent meaning (as in the case of idioms). Analogously, modifiers of a head verb (or event class) include its complements. Adjective and prepositional phrase IRules specify the semantic class of the nouns they can modify.

Also, maintenance facilities were added, as discussed in sections 4.3, 4.4, and 5.

4.1 IRules

An IRule defines, for a particular word or (semantic) class of words, the semantically acceptable English phrases that can occur having that word as head of the phrase, and in addition defines the semantic interpretation of an accepted phrase. Since semantic processing is integrated with syntactic processing in IRUS, the IRules serve to block a semantically anomalous phrase as soon as it is proposed by the parser. Thus, selectional restrictions (or case frame constraints) are continuously applied. However, the semantic representation of a phrase is constructed only when the phrase is believed complete.

There are IRules for four kinds of heads: verbs, nouns, adjectives, and prepositions. The left hand side of the IRule states the selectional restrictions on the modifiers of the head. The right hand side specifies the predicates that should be used in constructing a logical form corresponding to the phrase which fired the IRule.

When a head word of a phrase is proposed by the parser to the semantic interpreter, all IRules that can apply to the head word for the given phrase type are gathered as follows: for each semantic property that is associated with the word, the IRules associated with the given domain model term are retrieved, along with any inherited IRules. A word can also have IRules fired directly by it, without involving the model. Since the IRules corresponding to the different word senses may give rise to separate interpretations, they are carried along in parallel as the processing continues. If no IRules are retrieved, the interpreter rejects the word.

One use of the domain model is that of IRule inheritance. When an IRule is defined, the user decides whether the new IRule (the *base* IRule) should inherit from IRules attached to higher domain model terms (the *inherited* IRules), or possibly inherit from other IRules specified by the user. When a modifier of a head word gets transmitted and no pattern for it exists in a base IRule for the head word, higher IRules are searched for the pattern. If a pattern does exist for the modifier in a given IRule, no higher ones are tried even if it does not pass the semantic test. That is, inheritance does not relax semantic constraints.

4.2 An IRACQ session

In this section we step through the definition of a clause IRule for the word "send", and assume that lexical information about "send" has already been entered. The sense of "sending" we will define, when used as the main verb of a clause, specifies an event type whose representation is as follows:

(λ x) [deployment(x) & agent(x, a) & object(x, o) & destination(x, d)],

where the agent *a* must be a commanding officer, the object *o* must be a unit and the destination *d* must be a region.

From the example clauses presented by the user IRACQ must learn which unary and binary predicates are to be used to obtain the representation above. Furthermore, IRACQ must acquire the most general semantic class to which the variables *a*, *o*, and *d* must belong.

Output from the system is shown in bold face, input from the user in regular face, and comments are inserted in italics.

Word that should trigger this IRule: send

Domain model term to connect IRule to (select-K to view the network): deployment

> **<A:** *At this point the user may wish to view the domain model network using our graphical displaying and editing facility KREME [1] to decide the correct concept that should be associated with this word (KREME may in fact be invoked at any time). The user may even add a new concept, which will be tagged with the user's name and date for later verification by the domain model builder, who has full knowledge of the implications that adding a concept may have on the rest of the system.*
>
> *Alternatively, the user may omit the answer for now; in that case, IRACQ can proceed as before, and at* **B** *will present a menu of the concepts it already knows to be consistent with the example phrases the*

user provides. Figure 1 shows a picture of the network around DEPLOYMENT.>

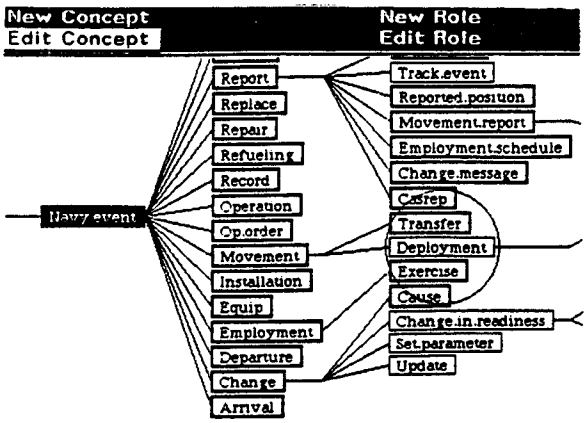

Figure 1: Network centered on DEPLOYMENT

Enter an example sentence using "send":
An admiral sent Enterprise to the Indian Ocean.

<IRACQ uses the full power of the IRUS parser and interpreter to interpret this sentence. A temporary IRule for "send" is used which accepts any modifier (it is assumed that the other words in the sentence can already be understood by the system.) IRACQ recognizes that an admiral is of the type COMMANDING.OFFICER, and displays a menu of the ancestors of COMMANDING.OFFICER in the NIKL taxonomy (figure 2).>

Choose a generalization for COMMANDING.OFFICER

COMMANDING.OFFICER
PERSON
CONSCIOUS.BEING
ACTIVE.ENTITY
OBJECT
THING

Figure 2: Generalizations of COMMANDING.OFFICER

<The user's selection specifies the case frame constraint on the logical subject of "send". The user picks COMMANDING.OFFICER. IRACQ will perform similar inferences and present a menu for the other cases in the example phrase as well, asking each time whether the modifier is required or optional. Assume that the user selects UNIT as the logical object and REGION as the object of the preposition "to".>

<B: If the user did not specify the concept DEPLOYMENT (or some other concept) at point A above as the central concept in this sense of "sending", then IRACQ would compute those unary concepts c such that there are binary predicates relating c to each case's constraint, e.g., to COMMANDING.OFFICER, REGION, and UNIT. The user would be presented with a menu of such concepts c. IRACQ would now proceed in the same way for A or B.>

<IRACQ then looks in the NIKL domain model for binary predicates relating the event class (e.g., DEPLOYMENT) to one of the cases' semantic class (e.g. REGION), and presents the user with a menu of those binary predicates (figure 3). Mouse options allow the user to retrieve an explanation of how a predicate was found, or to look at the network around it. The user picks DESTINATION.OF.>

Which of the following predicates should relate DEPLOYMENT to REGION in the MRL?:

LOCATION.OF
DESTINATION.OF

Figure 3: Relations between DEPLOYMENT and REGION

<IRACQ presents a menu of binary predicates relating DEPLOYMENT and COMMANDING.OFFICER, and one relating DEPLOYMENT and UNIT. The user picks AGENT and OBJECT, respectively.>

Enter examples using "send" or <CR> if done:

<The user may provide more examples. Redundant information would be recognized automatically.>

Should this IRule inherit from higher IRules? yes

<A popup window allowing the user to enter comments appears. The default comment has the creation date and the user's name.>

This is the IRule you just defined:

```
(IRule DEPLOYMENT.4
 (clause subject (is-a COMMANDING.OFFICER)
       head * object (is-a UNIT)
       pp ((pp head to pobj (is-a REGION))))
 ==>
 (bind ((commanding.officer.1 (optional subject))
        (unit.1 object)
        (region.1 (optional (pp 1 pobj))))
   (predicate '(destination.of *v* region.1))
   (predicate '(object.of *v* unit.1))
```

(predicate '(agent *v* commanding.officer.1))
(class 'DEPLOYMENT)))

Do you wish to edit the IRule? no

<The person may, for example, want to insert something in the action part of the IRule that was not covered by the IRACQ questions.>

This concludes our sample IRACQ session.

4.3 Debugging environment

The facility for creating and extending IRules is integrated with the IRUS NLI itself, so that debugging can commence as soon as an addition is made using IRACQ. The debugging facility allows one to request IRUS to process any input sentence in one of several modes: asking the underlying system to fulfill the user request, generating code for the underlying system, generating the semantic representation only, or parsing without the use of semantics (on the chance that a grammatical or lexical bug prevents the input from being parsed). Intermediate stages of the translation are automatically stored for later inspection, editing, or reuse.

IRACQ is also integrated with the other acquisition facilities available. As the example session above illustrates, IRACQ is integrated with KREME, a knowledge representation editing environment. Additionally, the IRACQ user can access a dictionary package for acquiring and maintaining both lexical and morphological information.

Such a thoroughly integrated set of tools has proven not only pleasant but also highly productive.

4.4 Editing an IRule

If the user later wants to make changes to an IRule, he/she may directly edit it. This procedure, however, is error-prone. The syntax rules of the IRule can easily be violated, which may lead to cryptic errors when the IRule is used. More importantly, the user may change the semantic information of the IRule so that it no longer is consistent with the domain model.

We are currently adding two new capabilities to the IRule editing environment:

1. A tool that uses some of the same IRACQ software to let the user expand the coverage of an IRule by entering more example sentences.
2. In the case that the user wants to bypass IRACQ and modify an IRule, the user will be placed into a restrictive editor that assures the syntactic integrity of the IRule, and verifies the semantic information with the domain model.

5 An IRule Paraphraser

An IRule paraphraser is being implemented as a comprehensive means by which an IRACQ user can observe the capabilities introduced by a particular IRule. Since paraphrases are expressed in English, the IRule developer is spared the details of the IRule internal structure and the meaning representation. The IRule paraphraser is useful for three main purposes: expressing IRule inheritance so that the user does not redundantly add already inherited information, identifying omissions from the IRule's linguistic pattern, and verifying IRule consistency and completeness. This facility will aid in specifying and maintaining correct IRules, thereby blocking anomalous interpretation of input.

5.1 Major design features

The IRule paraphraser makes central use of the IRUS paraphraser (under development), which paraphrases user input, particularly in order to detect ambiguities. The IRUS paraphraser shares in large part the same knowledge bases used by the understanding process, and is completely driven by the IRUS meaning representation language (MRL) used to represent the meaning of user queries. Given an MRL expression for an input, the IRUS paraphraser first transforms it into a syntactic generation tree in which each MRL constituent is assigned a syntactic role to play in an English paraphrase. The syntactic roles of the MRL predicates are derived from the IRules that could generate the MRL.

In the second phase of the IRUS paraphraser, the syntactic generation tree is transformed into an English sentence. This process uses an ATN grammar and ATN interpreter that describes how to combine the various syntactic slots in the generation tree into an English sentence. Morphological processing is performed where necessary to inflect verbs and adjectives, pluralize nouns, etc.

The IRule paraphraser expresses the knowledge in a given IRule by first composing a stereotypical phrase from the IRule linguistic pattern (i.e., the left hand side of the IRule). For the "send" IRule of the previous section, such a phrase is "A commanding officer sent a unit to a region". For inherited IRules, the IRule paraphraser composes representative phrases that match the combined linguistic patterns of both the local and the inherited IRules. Then, the IRUS parser/interpreter interprets that phrase using the given IRule, thus creating an MRL expression. Finally, the IRUS paraphraser expresses that MRL in English.

Providing an English paraphrase from just the linguistic pattern of an IRule would be simple and uninteresting. The purpose of obtaining MRLs for representative phrases and using the IRUS paraphraser to go back to the English is to force the use of the right hand side of the IRule which specifies the semantic

interpretation. In this way anomalies introduced by, for example, manually changing variable names in the right hand side of the IRule (which point to linguistic constituents of the left hand side), can be detected.

5.2 Role within IRACQ

IRACQ will invoke the IRule Paraphraser at two interaction points: (1) at the start of an IRACQ session when the user has selected a concept to which to attach the new IRule (paraphrasing IRules already associated with that concept shows the user what is already handled--a new IRule might not even be needed), and (2) at the end of an IRACQ session, assisting the user in detecting anomalies.

The planned use of the IRule Paraphraser is illustrated below with a shortened version of an IRACQ session.

Word that should trigger this IRule: change

Domain model term to connect IRule to:
change.in.readiness

Paraphrases for existing IRules (inherited phrases are capitalized):

Local IRule: change.in.readiness.1
"A unit changed from a readiness rating to a readiness rating"

Inherited IRule: event.be.predicate.1
"A unit changed from a readiness rating to a readiness rating"
{IN, AT} A LOCATION

<Observing these paraphrases will assist the IRACQ user in making the following decisions:

- *A new CHANGE.IN.READINESS.2 Irule needs to be defined to capture sentences like "the readiness of Frederick changed from C1 to C2".*
- *Location information should not be repeated in the new CHANGE.IN.READINESS.2 Irule since it will be inherited.*

The IRACQ session proceeds as described in the previous example session.>

6 Concluding Remarks

Our approach to semantic knowledge acquisition: 1) is in the context of a general purpose NL interface rather than one that accesses only databases, 2) employs a knowledge representation formalism with limited inferencing capabilities, 3) assumes a trained person but not an AI expert, and 4) provides a complete environment for not only acquiring semantic knowledge, but also maintaining and editing it in a consistent knowledge base. This section comments on what we have learned thus far about the point of view espoused above.

First, we have transferred the IRUS natural language interface, which includes IRACQ, to the staff of the Naval Ocean Systems Center. The person in charge of the effort at NOSC has a master's degree in linguistics and had some familiarity with natural language processing before the effort started. She received three weeks of hands-on experience with IRUS at BBN in 1985, before returning to NOSC where she trained a few part-time employees who are computer science undergraduates. Development of the dictionary and IRules for the Fleet Command Center Battle Management Program (FCCBMP), a large Navy application [23], has been performed exclusively by NOSC since August, 1986. Currently, about 5000 words and 150 IRules have been defined.

There are two strong positive facts regarding IRACQ's generality. First, IRUS accesses both a large relational data base and an applications package in the FCCBMP. Only one set of IRules is used, with no cleavage in that set between IRules for the two applications. Second, the same software has been useful for two different versions of IRUS. One employs MRL [29], a procedural first order logic, as the semantic representation of inputs; the second employs IL, a higher-order intensional logic. Since the IRules define selectional restrictions, and since the Davidson-like representation (see section 3) is used in both cases, IRACQ did not have to be changed; only the general procedures for generating quantifiers, scoping decisions, treatment of tense, etc. had to be revised in IRUS. Therefore, a noteworthy degree of generality has been achieved.

Our key knowledge representation decisions were the treatment of events and states of affairs, and the use of NIKL to store and reason about axioms concerning the predicates of our logic. This strongly influenced the style and questions of our semantic acquisition process. For example, IRACQ is able to propose a set of predicates that is consistent with the domain model to use for the interpretation of an input phrase. We believe representation decisions must dictate much of an acquisition scenario no matter what the decisions are. In addition, the limited knowledge representation and inference techniques of NIKL deeply affected other parts of our NLI, particularly in the translation from conceptually-oriented domain predicates to predicates of the underlying systems.

The system does provide an initial version of a complete environment for creating and maintaining semantic knowledge. The result has been very desirable compared to earlier versions of IRACQ and IRUS that did not have such debugging aids nor integration with tools for acquiring and maintaining the

domain model. We intend to integrate the various acquisition, consistency, editing, and maintenance aids for the various knowledge bases even further.

References

1. Abrett, G., and Burstein, M. H. The BBN Laboratories Knowledge Acquisition Project: KREME Knowledge Editing Environment. BBN Report No. 6231, Bolt Beranek and Newman Inc., 1986.

2. Allen, J.F. and Litman, D.J. "Plans, Goals, and Language". *Proceedings of the IEEE 74*, 7 (July 1986), 939-947.

3. Amsler, R.A. A Taxonomy for English Nouns and Verbs. Proceedings of the 19th Annual Meeting of the Association for Computational Linguistics, 1981.

4. Ballard, Bruce and Stumberger, Douglas. Semantic Acquisition in TELI: A Transportable, User-Customized Natural Language Processor. Proceedings of The 24th Annual Meeting of the ACL, ACL, June, 1986, pp. 20-29.

5. Bates, M. and Bobrow, R.J. A Transportable Natural Language Interface for Information Retrieval. Proceedings of the 6th Annual International ACM SIGIR Conference, ACM Special Interest Group on Information Retrieval and American Society for Information Science, Washington, D.C., June, 1983.

6. Bates, Madeleine. Accessing a Database with a Transportable Natural Language Interface. Proceedings of The First Conference on Artificial Intelligence Applications, IEEE Computer Society, December, 1984, pp. 9-12.

7. Bates, M., and Ingria, R. Dictionary Package Documentation. Unpublished Internal Document, BBN Laboratories.

8. Carberry, M.S. A Pragmatics-Based Approach to Understanding Intersentential Ellipsis. Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Chicago, IL, July, 1985, pp. 188-197.

9. Cumming, S, and Albano, R. A Guide to Lexical Acquisition in the JANUS System. Information Sciences Institute/RR-85-162, USC/Information Sciences Institute, 1986.

10. Davidson, D. The Logical Form of Action Sentences. In *The Logic of Grammar*, Dickenson Publishing Co., Inc., 1975, pp. 235-245.

11. Granger, R.H. "The NOMAD System: Expectation-Based Detection and Correction of Errors during Understanding of Syntactically and Semantically Ill-Formed Text". *American Journal of Computational Linguistics 9*, 3-4 (1983), 188-198.

12. Grishman, R. Hirschman, L., and Nhan, N.T. "Discovery Procedures for Sublanguage Selectional Patterns: Initial Experiments". *Computational Linguistics 12*, 3 (July-September 1986), 205-215.

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

14. Grosz, B.J. and Sidner, C.L. Discourse Structure and the Proper Treatment of Interruptions. Proceedings of IJCAI85, International Joint Conferences on Artificial Intelligence, Inc., Los Angeles, CA, August, 1985, pp. 832-839.

15. Litman, D.J. Linguistic Coherence: A Plan-Based Alternative. Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics, ACL, New York, 1986, pp. 215-223.

16. Markowitz, J., Ahlswede, T., and Evens, M. Semantically Significant Patterns in Dictionary Definitions. Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics, June, 1986.

17. Mooney, R. and DeJong, G. Learning Schemata for Natural Lanugage Processing. Proceedings of the Ninth International Joint Conference on Artificial Intelligence, IJCAI, 1985, pp. 681-687.

18. Moser, M.G. An Overview of NIKL, the New Implementation of KL-ONE. In *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, Sidner, C. L., et al., Eds., BBN Laboratories Report No. 5421, 1983, pp. 7-26.

19. Moser, M. G. Domain Dependent Semantic Acquisition. Proceedings of The First Conference on Artificial Intelligence Applications, IEEE Computer Society, December, 1984, pp. 13-18.

20. Schank, R., and Abelson, R. *Scripts, Plans, Goals, and Understanding.* LawrenceErlbaumAssociates, 1977.

21. Schmolze, J. G., and Israel, D. J. KL-ONE: Semantics and Classification. In *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, Sidner, C.L., et al., Eds., BBN Laboratories Report No. 5421, 1983, pp. 27-39.

22. Sidner, C.L. "Plan Parsing for Intended Response Recognition in Discourse". *Computational Intelligence 1*, 1 (February 1985), 1-10.

23. Simpson, R.L. "AI in C3, A Case in Point: Applications of AI Capability". *SIGNAL, Journal of the Armed Forces Communications and Electronics Association 40*, 12 (1986), 79-86.

24. Stallard, D. Data Modelling for Natural Language Access. The First Conference on Artificial Intelligence Applications, IEEE Computer Society, December, 1984, pp. 19-24.

25. Stallard, David G. A Terminological Simplification Transformation for Natural Language Question-Answering Systems. Proceedings of The 24th Annual Meeting of the ACL, ACL, June, 1986, pp. 241-246.

26. Vilain, M. The Restricted Language Architecture of a Hybrid Representation System. Proceedings of IJCAI85, International Joint Conferences on Artificial Intelligence, Inc., Los Angeles, CA, August, 1985, pp. 547-551.

27. Walker, E., Weischedel, R.M., and Ramshaw, L. "IRUS/Janus Natural Language Interface Technology in the Strategic Computing Program". *Signal 40*, 12 (August 1986), 86-90.

28. Weischedel, R.M. and Ramshaw, L.A. Reflections on the Knowledge Needed to Process Ill-Formed Language. In *Machine Translation: Theoretical and Methodological Issues*, S. Nirenburg, Ed., Cambridge University Press, Cambridge, England, to appear.

29. Woods, W.A. Semantics and Quantification in Natural Language Question Answering. In *Advances in Computers*, M. Yovits, Ed., Academic Press, 1978, pp. 1-87.