Application of Natural Language Interface to a Machine Translation Problem

Heidi M. Johnson
Yukiko Sekine
John S. White
Martin Marietta Corporation

Gil C. Kim
Korean Advanced Institute of
Science and Technology

Abstract. Issues revolving around transportability and technology transfer have been of growing interest to natural language processing development. This paper describes an experiment performed to test the modularity of the Martin Marietta EQUAL natural language processor, by adapting this database interface system to English-Korean machine translation task. The methodology included altering the application interface module to convert the intermediate representation of the EQUAL output into the input expected by a Korean generator. The treatment of general problems concerning the nature of interlingua-transfer strategies, and particular problems of English-Korean transfer is presented.

O. Introduction. The increase in the ability to develop usable natural language technologies has led to an interest, among the applications-oriented development communities, in integrating these natural language capabilities to perform other computational tasks. Many natural language systems, however, were designed either to serve a particular research end, or a narrowly-focussed commercial end, the result in both cases being a relative lack of ability to transport their functionalities to other uses. The following is a description of a recent effort by Martin Marietta Data Systems, in conjunction with the Korean Advanced Institute for Science and Technology, to apply a natural language processor originally designed for database interface to a machine translation problem. This effort demonstrates the ability to link such a natural language processor to a Korean-made natural language processor, by developing an algorithm for converting the English intermediate representation into the intermediate representation expected by the Korean system. This exercise has demonstrated the usefulness of design oriented toward the sort of modularity that will become more in demand by those in the applied areas of computational linguistic work, namely.

The approach used in the experiment consisted of creating a set of transfer procedures which convey the necessary information output from the Martin Marietta system (known as EQUAL) into the abstract formulas which serve as the input to a proprietary Korean synthesizer developed by the Korean Advanced Institute of Science and Technology. By this means, an English sentence typed in by a user can be processed into an abstract form from which the Korean synthesizer can produce a Korean (Hangul) sentence.

This account includes a general description of the EQUAL technology, an account of machine translation approaches, the specific design approach taken for this application experiment, and the implementation methodology.

1.0. The EQUAL Natural Language Processor. The Martin Marietta Data Systems Natural Language Interface is a technology representing development efforts in natural language processing conducted since 1982. The present technology, a prototype aimed for intelligence and military markets, is a natural language parsing capability that has demonstrable interfaces to Ingres, Ramis, and M204 databases. Using EQUAL, the database user communicates with a DBMS directly in English instead of complicated query languages - particularly those associated with non-relational databases, but with the almost equally difficult "4GLs" as well.

The EQUAL prototype now under demonstration runs on Sun2 and Sun3 workstations, along with the Intelligence Workstation, a TEMPESTed, Unix-based workstation. System ports to other machines, such as Vaxes, 80386-based machines, and PS2's, are likely to be pursued in the coming year. In addition to the DBMSs mentioned above, interfaces to the traditional IMS database, and to the Sperry DMS-1100 database, are under development, using the same natural language processor.

The principal development thrusts for EQUAL, in addition to the task described in this paper, are speed enhancement, enhancement of linguistic coverage, and knowledge-based installation routines for the DB interface module.

EQUAL is designed in such a way as to be maximally modular, which enhances not only configuration management and maintenance, but also the ability to re-use modules of the system for other applications. The linguistic parts of the system (the dictionaries, the morphological handler, the syntax, and the semantics) are all separated from each other and

from the parser control structure. As a result, only the linguistic modules need to be replaced in order to parse a new language; the control software remains in place.

Similarly, the parser remains separate from both the input and output control structures. Thus, while the present EQUAL expects a user-machine dialogue as input, it can easily accept text inputs as well. The output control, driven by the application interface module, can be adapted to convert the parser output into the input required by different DBMSs, as has been done, or into source code for an automatic programming task, or as the input required by a natural language generator, as the present study shows. This adaptation is done without affecting either the natural language parser or the backend application software (in this case a Korean language generator).

There are similarities between the philosophy under which this experiment was undertaken, and other natural language efforts where transportability was involved (Gross 1983 , Hafner 1984). There are similarities as well with studies in which a language processor converted from one language to another (Lopes 1984), and in efforts involved with system integration (Boitet 1985). The transportability efforts usually involved a conversion for the purposes of the same overall functionality to other applications (different database domains, different DBMS products). In the case of system integration, different technologies were grafted in to be a part of the natural language processor itself. The EQUAL/VENUS experiment described here differs from these by being a re-use, for an entirely different application, of a natural language processor technology. In this regard, the effort seems much more like the various studies associated with Mumble (e.g., McDonald and Meteer 1988), in which different sorts of application program output was made to conform, via inferences about the information provided in that output, to the input expected by Mumble.

2.0. Technology Transfer Approach. The application of the existing Natural language processing system to the task of English/Korean translation proceeded from the hypothesis that it is feasible to write a connecting module from the intermediate representation produced by EQUAL into the input expected by a Korean natural language synthesizer. The successful solution of this problem automatically demonstrates the capability of EQUAL to perform machine translation: it is a difficult test, inasmuch as it must interface a NLP with a potentially radical difference in expectations than those produced by the present natural language database interface. Further, the Korean system to which it is to be interfaced (the Korean synthesizer for the VENUS machine translation system developed by KAIST for Nippon Electric; see Muraki 1984) is of a different fundamental design (an interlingua-type MT system) than is the resulting combination (a transfer-type MT system), and the natural language requirements of the VENUS project are different than required by this investigation (Japanese/Korean rather than English/Korean). Consequently, this task is conceptually much more indicative of the capabilities of the EQUAL natural language technology than would be a small prototype of a new development machine translation.

As mentioned above, the VENUS prototype is an interlingua-type MT system, while the model resulting from this effort is a transfer-type MT system. The differences concern the number and functions of the components of the system, as well as practical considerations of implementation within real-time requirements.

Interlingua. The criterion for interlingua is that the same device that interprets a language is used to generate that language (Slocum 1985); consequently, the output of the interpreter (parser) must be a general, universal, representation of the meaning of the input that allows synthesis of the same meaning into any language. The advantages of the design

are that the interfaces among the grammars and lexicons of different languages are completely uniform (i.e., only through the single abstract intermediate representation), and thus there need be only one grammar module for each language (which does both the job of parsing and generating). The disadvantages are that there are no grammar designs which are truly bi-directional (interpreters which resolve many-to-one problems cannot reverse the process in predictable ways), and there is no purely semantic representation which can capture all the information necessary to perform correct translation.

Transfer. The transfer strategy differs from the interlingua method in that the intermediate representation (or set of intermediate representations) are not simply expressions of meaning, but are an expression of a homogenized form which much of the syntactic information (the structure of the input sentence) is preserved. The transfer design generally has two such representations, one for the source language which is converted into one for the target language. The target language is synthesized in a very straightforward way from the TL intermediate representation, most of the conversion work having been done by transfer algorithms converting the SL intermediate into the TL intermediate. Consequently, the synthesis module is very much simpler than the interpretation module. While this approach increases the number of individual components, and is ultimately less powerful than interlingua designs, there is less overall processing required and more of the relevant information for translation is maintained. Transfer systems typically have transfer dictionary structures as well. These allow for individual dictionaries of the separate languages, containing only language-specific information, and a bilingual dictionary relating words to translations of those words. Such transfer dictionaries are often quite powerful, and capable of structural and meaning manipulations assisting the parser and generators.

The approach described here took the EQUAL system, which is neutral with respect to these design types, and interfaced it to an interlingua-type machine translation design, via a set of transfer algorithms. In effect, the VENUS interlingua becomes a TL transfer intermediate representation for the purposes of this experiment

Additionally, a device essentially the same as a transfer dictionary was employed, which, however, instead of transferring words, transfers word concepts (i.e, the metalinguistic symbol representation of the reference of the lexeme, rather than a canonical morpheme representing the name of the word). At present, transferring at the conceptual level rather than at the lexical level takes advantage of the fact that both the EQUAL and VENUS systems have fairly robust lexical semantic structures in place. This transfer dictionary further demonstrates the feasibility of adapting the existing systems into one transfer system.

Table 2.1 shows the basic plan of this proof for optimal technology re-use.
The shaded areas represent the modules which have been developed. The highlighted component is the set of transfer algorithms developed for the experiment.

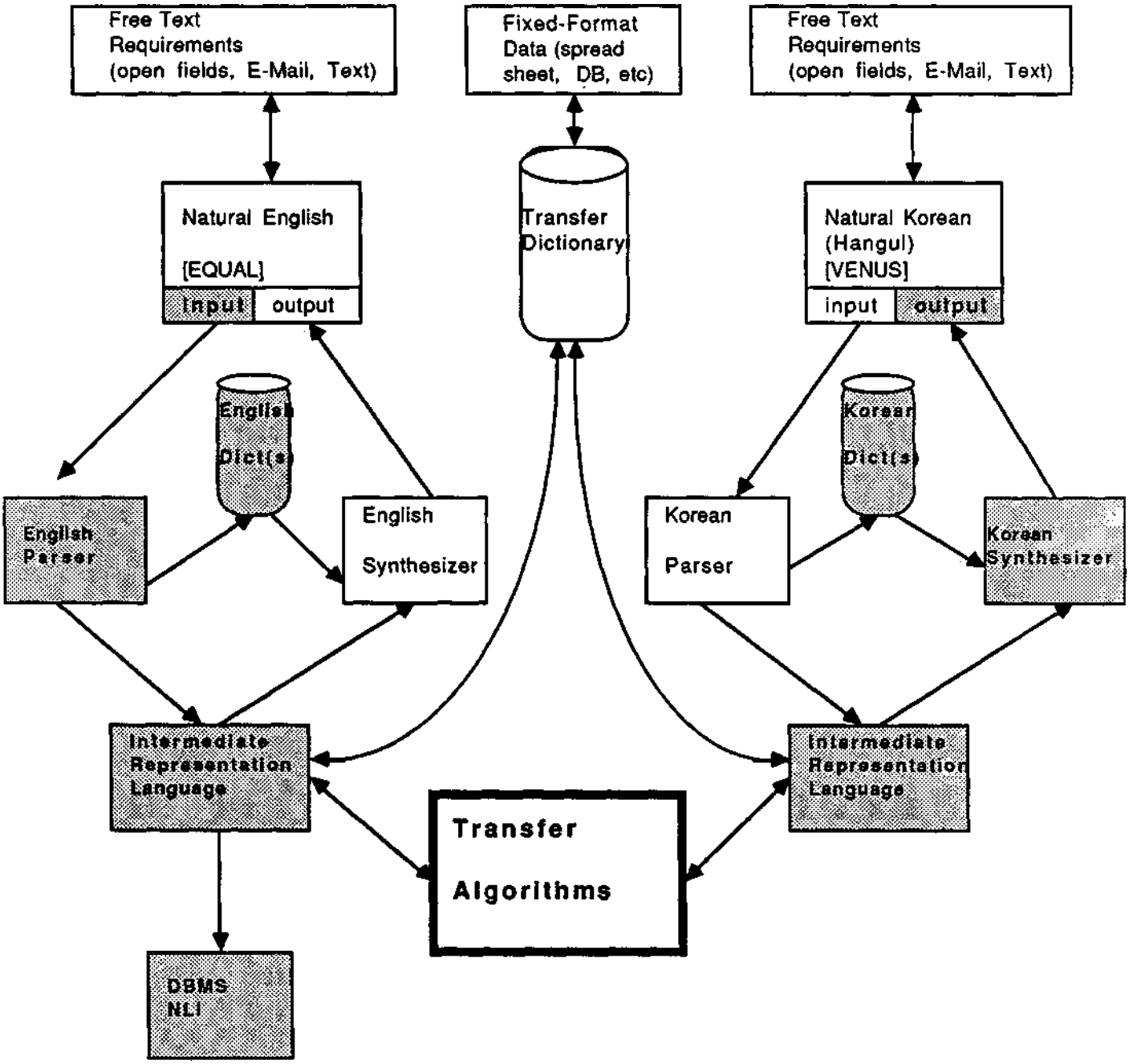

Table 2.1. EQUAL/VENUS Transfer Architecture

3.0. Design of the Transfer Module. The principle behind the transfer module involves exploiting the portability of the EQUAL design as a whole. At the conceptual level, a statement in one language and its equivalent in another language convey very similar information. Such information is independent of any particular NLP approach, natural language syntactic distributional requirements. Thus, the core of the module is the intermediate representation, which is isomorphic to the output of the parse of the source language and the input to the generator of the target language.

During the course of designing the intermediate representation, it was expected that some language-specific information must be taken into consideration. Some of the information, such as many-to-one and one-to-many mapping relations, can be handled by the individual language processors, but one-to-one relations should be handled by the transfer module. Thus, the transfer module requires the intermediate representation, interface submodules, a transfer dictionary, and a set of transfer rules. Except for the intermediate representation, the exact procedure for each of the above is determined in terms of the natural language processors employed, the differences between the source and target languages, the domain of the application, and the hardware and software available.

3.1. High-Level Parse Trees. The simple sentence "I use a high-speed computer" illustrates the approach. A very high-level conceptual representation of the parse result of EQUAL is shown in Fig. 3.1 (a). Note that the root node is "USE," the head of the predicate of the sentence.

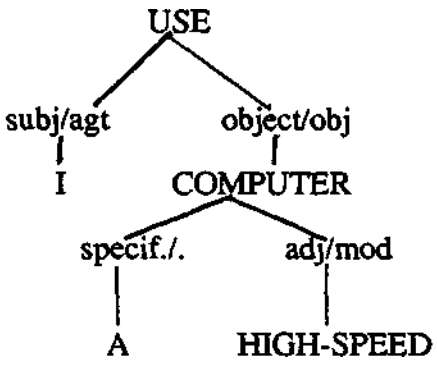

Figure 3.1a EQUAL parse.

Fig. 3.1 (b) below is the VENUS equivalent of the parse analysis at the same level:

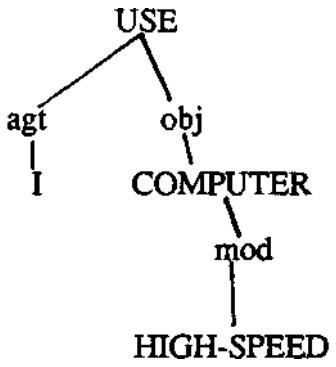

Figure 3.1.b. VENUS parse.

From the above figures, it is evident that the both processors produce similar parse results. There are, however, some differences between the two processors. For instance, EQUAL provides more detailed semantic information than VENUS does; VENUS uses more case roles than EQUAL; VENUS uses some functions (such as FEATURES) that are absent from EQUAL. Some other differences stem from the differences between English and Korean, including:

• lack of articles in Korean

• lack of honorifics in English

• use of explicit case markers in Korean

3.1.1. Lack of Articles in Korean. This problem may seem to be easy to handle when the language direction is English into Korean: all that must be done is to drop the determiner in the generated structure. It is not quite that easy, however. When specificity is not marked, the determiner can be dropped, but when it is marked, determiners in Korean must be assigned (except for idiomatic expressions and special conventions such as the names of rivers and newspapers, and generic concepts). When both the speaker and hearer know a third person, for example, they use Korean word for "that", but if only the speaker knows the person, the speaker must use "this", and otherwise the equivalent of "the". Therefore, information regarding specificity must be represented in the intermediate representation.

3.1.2. Lack of Honorifics in English. English can reflect some subtle differences in deference and formality, but such differences are not built in explicitly in its grammar. Korean, on the other hand, maintains very explicit linguistic distinctions for different levels of formality and politeness. Since the information about the level of honorifics can only be made in Korean, for the purposes of this experiment the POLITE mode was assigned as the default value.

3.1.3. Use of Explicit Case Markers in Korean. Case roles are explicitly marked by post position particles in Korean. The same information can be represented in terms of word order and/or prepositional phrases in English. There are some case roles that are shared by both English and Korean, and there are others that are not. For example, INSTRUMENT, LOCATIVE, and TIME have a one-to-one mapping between the two

languages. OBJECT in English, on the other hand, is marked by one of two particles *e* and *rul* in Korean, (corresponding to the OBJECT and TARGET cases, respectively, in the VENUS model), depending on the particle a verb takes (e.g., "BECOME," "ANSWER, "RESPONSE"). AGENT in English is also marked by one of two particles, *ga* and *nun,* depending on the focus: if AGENT is new information, it is marked by *ga;* if the AGENT is old information, it is marked by *nun,* the topic marker (corresponding to the AGENT and REASON cases employed in VENUS).

3.1. The Intermediate Representation. The intermediate representation is a Lisp-readable parse tree generated from the output of the EQUAL parser. It includes features and modifiers that are independent of the specific back-end application; these features (in the form of frames containing slot-value pairs) consist of the attributes (syntactic and semantic) of sentence constituents recognized as necessary and relevant for a felicitous determination of syntactic and semantic dependencies. The lists are stored in global variables that are easily altered to accommodate variations in the target generation system and instrument. The algorithm begins at the root (top) node of the EQUAL output parse and recursively pursues pointers to the case fillers and modifiers of each subtree. The general form of the tree is:

```
(0 (root-concept root-word)
  (<root-syntactic-features>)
  ((level (case name
          (level+1 (case-concept case-word)
                   (<case-syntactic-features>)
                   (<case-semantic-cases>))))))
```

The concept names are the names of frames in the EQUAL knowledge base; the case names are the names of slots on those frames that have values for the given parse. If there are no relevant features or cases for a given word, the place is indicated by a nil value. The parse tree resulting from the sentence "I use a high-speed computer" is:

```
(0 (use use)
  ((tense present)(aspect finite)(voice active)(number singular))
  ((1 (agent (2 (human I)((number singular)) nil)))
  (1 (object (2 (computer computer)
                ((number singular)(specificity indefinite))
                ((3 dimension
                (4 (temporal high-speed) nil nil))))))))
```

Table 3.1 shows the overall control flow of the transfer algorithm. Tables 3.2 and 3.3 are flow charts representing the creation of the normalized EQUAL parse tree, and the concept transfer tasks, respectively.

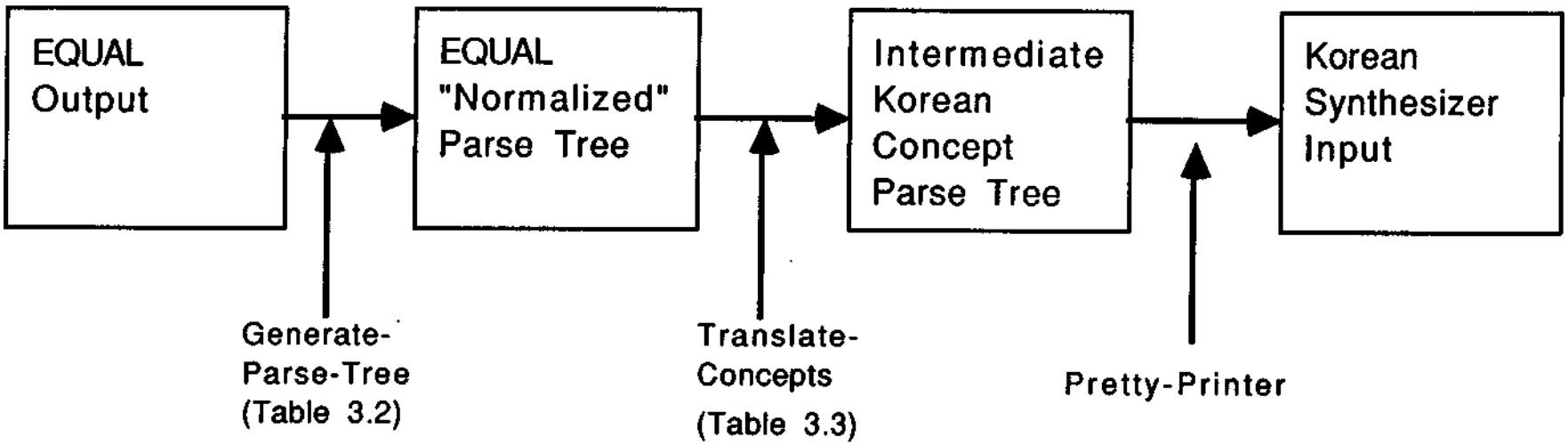

Table 3.1. Control Structure of Transfer Module

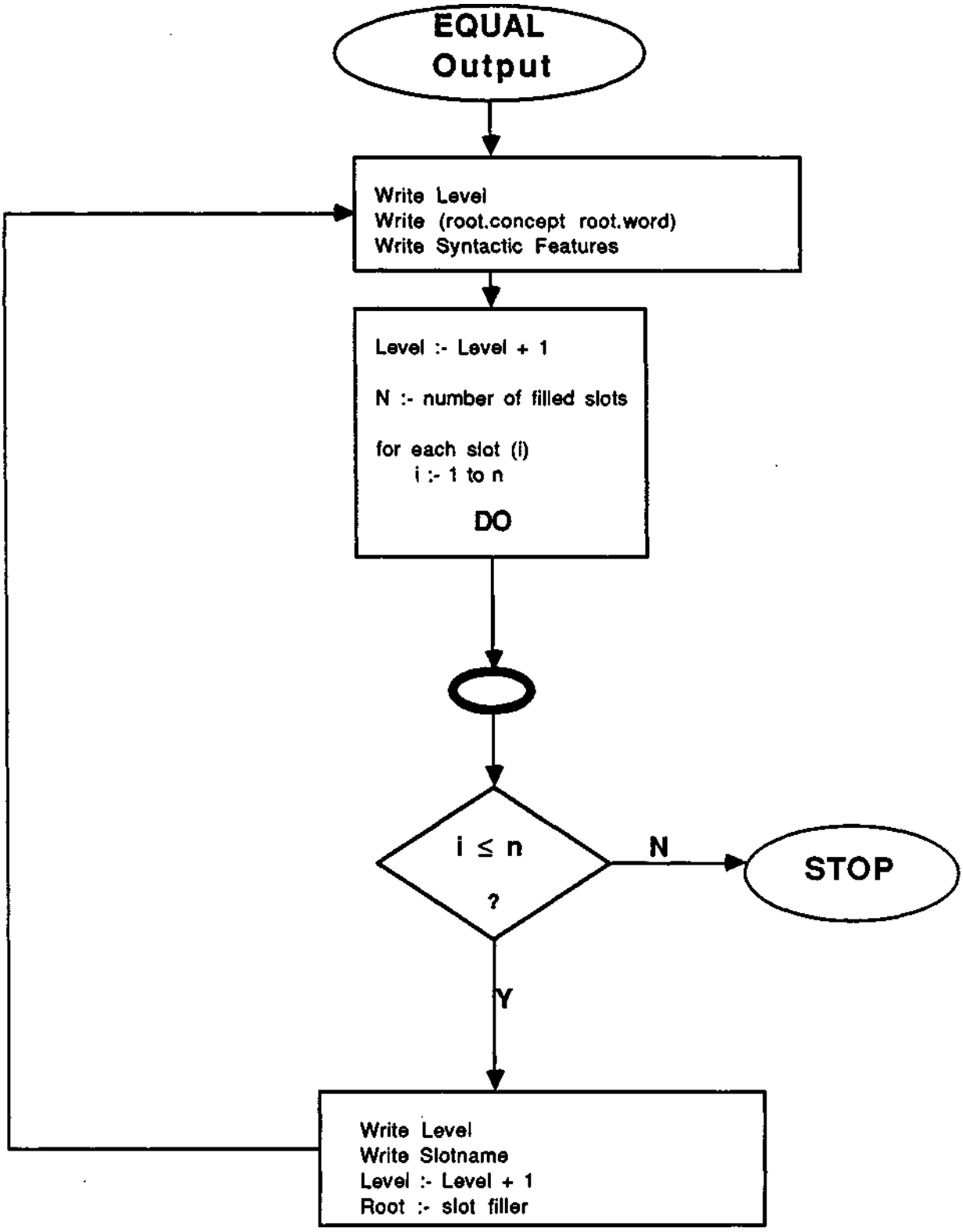

Table 3.2. Generate-Parse-Tree.

3.3. Transfer Algorithm. The EQUAL parse tree is used as input to the translation module, which converts it into a similar, Lisp-readable parse tree that uses the concept and feature names of the VENUS system. The conversion is directed by the information in the concept transfer dictionary, which contains entries for many of the EQUAL frames, case names and features. If an entry is not found for a given frame, the root word is used, in keeping with the strategy evidently employed by the VENUS system.
A dictionary entry conforms to the following specification:

<entry> ==> (<EQUAL-frame> (<concept-translation>) (<feature-specification>))

<EQUAL-frame> ==> the name of a frame in the EQUAL hierarchy

<concept-translation> ==> { (VENUS-concept) |
(check-root (root-word (<root-translation>))) |
(check-mods (<slot-name> (<mod-translation>))) }

<root-translation> ==> { (VENUS-concept) |

(check-mods (<slot-name> (<mod-translation>)))}

<mod-translation> ==> (VENUS-concept)

<slot-name> ==> slot on the EQUAL frame

<feature-specification> ==> { (<VENUS-feature-pair>+) |
(check-root (root-word (<VENUS-feature-pair>+))) }

<VENUS-feature-pair> ==> (feature-name feature-value)

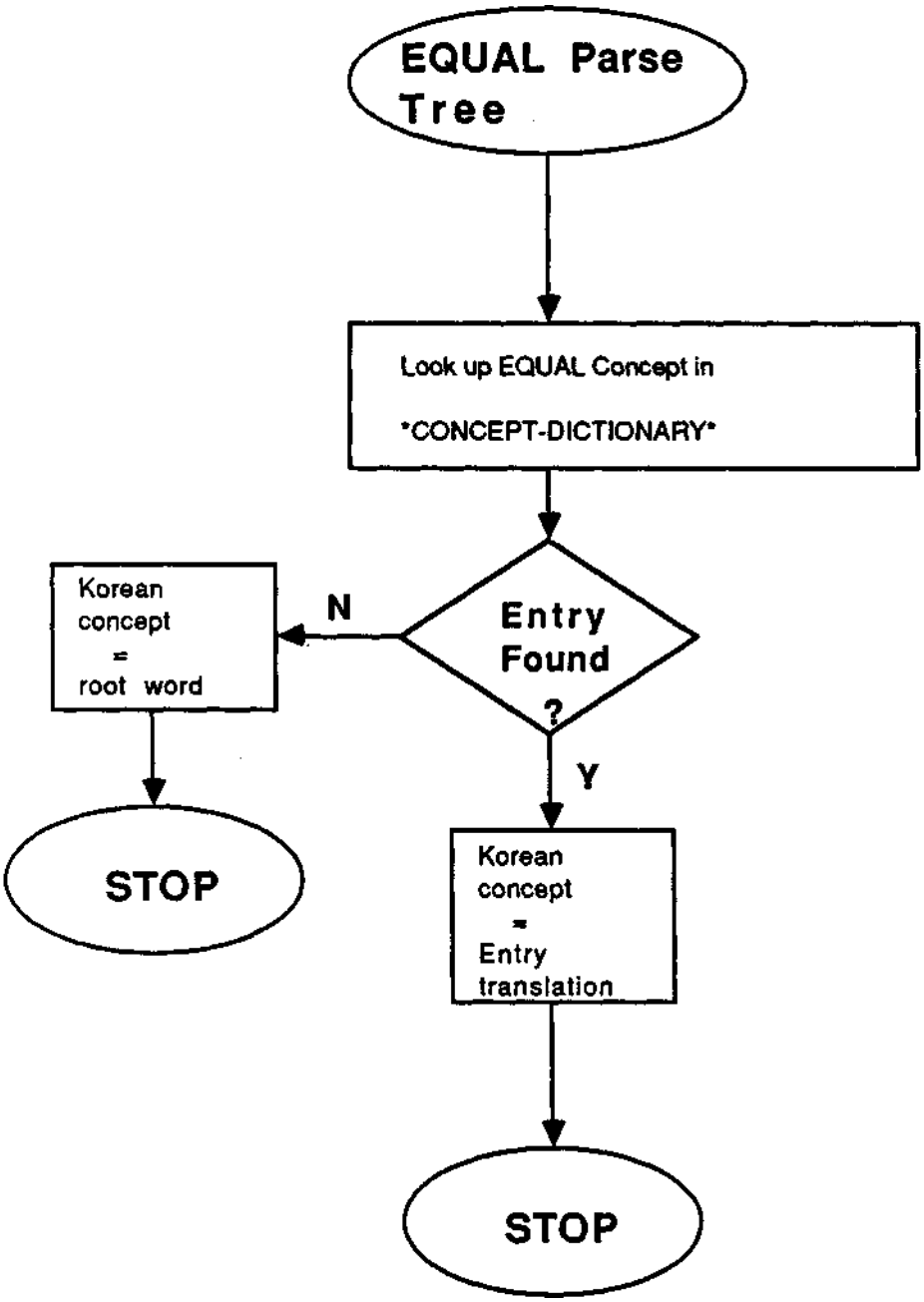

Table 3.3. Translate-Concepts

The flags "check-root" and "check-mods" are used to differentiate between similar concepts whose translations depend on the exact word used, or the modifiers present in the context. The output of the transfer module for the example sentence "I use a high-speed computer" is:

```
(0 ((CP USE)(STYLE DECL,POLIT)(TENSE PRE)(CFRAME (AGENT OBJECT
GOAL)))
((1 (CASE AGT)
   (2((CP I))))
   (1 (CASE OBJ)
   (2 ((CP COMPUTER))
   (3 (CASE MOD)
   (4 ((CP HIGH^SPEED))))))))
```

The extra step involved in generating this second parse tree was deemed worthwhile because it is immediately machine-readable and therefore amenable to any number of input formalism requirements. In the current implementation, the tree is passed to a pretty-printer that produces the input expected by the VENUS parser, and writes it to the file 'translations. The final result of the translation process for the example sentence is:

```
{0} #CP.{USE} #EXPRESSION.{POLIT} #STYLE.{DECL> #TENSE.{PRE}
        #CFRAME.{AGT, OBJ, GOA}
|
|==>{1}#CASE.{AGT)
|   |
|   |==>{2} #CP.{I}
|==>{1}#CASE.{OBJ}
|   |
|   |==>{2} #CP.{COMPUTER}
|   |   |
|   |   |==>{3} #CASE.{MOD}
|   |   |   |
|   |   |   |==>{4} #CP. {HIGH-SPEED}
```

4.0. Conclusion. The impetus for this experiment was driven by an orientation to application development in natural language processing. The Korean synthesizer developed for the VENUS project was not chosen for the experiment because of its similarity to EQUAL; this choice was driven by more mundane, industry considerations. Yet, it happens that the output of EQUAL is similar to the input to VENUS, with respect to the semantic/conceptual primacy of the data represented (EQUAL) and employed (VENUS).

Even here, though, the differences between the languages themselves will form significant challenges to the approach, should it be extended into genuine application. Differences in the handling of deference, determiner presence and scope, and marking of case information (which may not be a one-to-one map to the markings of the other language), will affect the means by which the intermediate representation must be manipulated. The methodology for this manipulation involves an extension to the expert system already employed in the natural language interface application of EQUAL.

This experiment has served to demonstrate the re-use potential of a particular set of natural language processors. In so doing, it has illustrated the value of employing natural language designs in the application world with a view toward the larger vision of application, namely, by building in the modularity which allows extension not only to the immediate application requirement, nor to those visible today, but also to those application needs not yet articulated.

References

Gerber, R., and C. Boitet. 1985. "On the Design of Expert Systems Grafted on MT Systems." *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages,* pp. 116-134.

Gross, B. 1983. "TEAM, a Transportable Natural Language Interface." *Proceedings of the Conference on Applied Natural Language Processing,* pp. 39-45.

Hafner, C.D. 1984. "Interaction of Knowledge Sources in a Portable Natural Language Interface." *Proceedings of COLING84* p.57-60.

Lopes, G.P. 1984. "Transforming English Interfaces to Other Natural Languages: An Experiment with Portugese." Proceedings of Coling84. pp.8-10.

McDonald, D. and M. Meteer. 1988. "From Water to Wine: Generating Natural Language Text From Today's Applications Programs." *Proceedings of the Second Conference on Applied Natural Language Processing.* Association for Computational Linguistics. pp.41-48.

Muraki, K. 1984. "Japanese-English Machine Translation System Using Knowledge Base and Independent Intermediate Expressions." *Nikkei Electronics,* vol. 17.2 (in Japanese; title translated).

Slocum, J. 1985. "A Survey of Machine Translation: its History, Current Status, and Future Prospects." *Computational Linguistics* vol. 11.1. pp. 1-17.