A MULTILINGUAL ARCHITECTURE FOR DATABASE ACCESS AND MACHINE TRANSLATION

Juan Carlos Martínez Guerra

MCC
3500 West Balcones Center Drive
Austin, Texas 78759 USA

1.- Multilingual Computing

A.- Multilingual access to distributed information resources

In today's corporate world, with trends towards global markets and internationalization, there is an increasing need to access, share, and communicate diverse sources of information involving different languages. The diversity in the sources comes from different types of applications - documents, database management systems, repositories of information, spreadsheets, etc. Multilingual computing means different things depending on the intended use of the sources of information as well as the different types of applications. For documents or flat repositories of facts, intended uses could be to access or index the information according to its content or to translate the contents of it. For databases, possible uses would be to store, retrieve or update the information from the database according to the fields in which the data was structured. Multilingual systems allow all of these operations to be done in any natural language.

The purpose of this paper is to describe a natural language (NL) system architecture for two main applications: machine translation and multilingual natural language access to databases. In order to address the issue of whether it makes sense to have a single architecture, we will briefly present the driving goals for each of the two applications. Then we will argue for a common architecture on the basis of the shared linguistic and domain knowledge used in both tasks. That shared knowledge is used in the system to produce a representation of the literal meaning of the input, which is the same irrespective of the application. That representation is then interpreted in the context of the intended use, and will vary from application to application. For machine translation (MT), the goal is to produce a natural sounding translation in the target language. As a front end system for databases, the aim is to produce a command to manipulate or describe information about the database(s). We then conclude this paper with a status report on the existing system and future work.

B.- Two distinct NL applications: MT and database interfaces

We would like to have a general purpose multilingual natural language system that could be used for a variety of applications. There are NL systems that have been used for multiple applications, but in all those applications the intended use of the system is to produce as output a command(s) to extract information from multiple underlying sources of information (i.e., spreadsheets, databases, etc.) [Bobrow, Resnik, Weischedel, 1990]. We have chosen two applications: machine translation and front-end systems for databases, which are sufficiently distinct in nature to demonstrate the feasibility of using a general system for two very different tasks. As a front-end system, the intended use is to produce a command(s) to access heterogeneous, distributed resources. For machine translation, however, the intended use is to produce a natural translation of the input.

We would like to use the same Natural Language system to do both multilingual retrieval from databases as well as machine translation. Before addressing the issue of whether it makes sense to have a common single architecture for the two tasks, let us look first at the driving goals for each of the two applications.

1.- Machine Translation (MT)

In [Barnett et al., 1990b], [Barnett et al., 1991b] [Barnett, Mani, Rich, 1991d] we describe an approach to machine translation motivated by the following two goals: first, the ability of the system to produce correct and natural sounding translations in the case where different explicit propositions are required in the target language than in the source, and second, to minimize the cost of adding a new language to the system.

The first of our goals is motivated by the divergences and mismatches that occur in translation. Translation divergence [Dorr, 1990] occurs when the same information is conveyed in the source and target text but the sentence structures are different. For example: «They hammered down the wall» and the Spanish equivalent: «Tumbaron la pared a martillazos». Translation mismatches [Kameyama, et al., 1991] occur when the information conveyed is in fact different in the two languages. The difference lies in the fact that the source text has ambiguity that is not allowed in the target language, in which case information needs to be added by making the best possible guess about the intent of the source text before rendering it to the target text. For example, in translating from English to Japanese it is necessary to add information about politeness. Similarly, information that is present in the source text but not necessarily required in the target language would have to be discarded when translating. An example of translation mismatch at the lexical level would be the English word «fish» which has two translations in Spanish depending on whether the fish is caught and fit for food («pescado») or not («pez»). So again information needs to be added when translating from English to Spanish, and discarded conversely.

Based on those stated goals we make the observation that transfer-based MT systems, where transfer rules between specific structures of pairs of languages are used, fall short for two reasons. The first shortcoming is obvious and has to do with the fact that since the mappings are done between language pairs, the cost of adding new languages is large. The second reason is that there are mismatch examples that cannot be handled effectively by the transfer-based systems. Their limitation resides in the fact that they are confined exclusively to the linguistic information present in the source text, but do not have a meaning representation with which to reason about and from which to derive more information not present explicitly in the source text. For example, transfer-based systems could handle the correct translation of «fish» in «Mary cooked fish» with simple type checking, however, most of them cannot differentiate between the following two sentences: «There were five fish in the lake» and «There were five fish in the casserole» (Taken from [Barnett et al., 1991b]) in a way that would produce «pez» in the first and «pescado» in the second. Thus, in order to achieve the stated goals, it is necessary for the system to produce a meaning-based representation of the content of the source text. That representation can be used as a starting point to infer the required information not explicitly mentioned in the source text from the discourse context and a model of the domain being discussed.

2.- NL access to databases

The design of an NL system as a front-end to a database is motivated by the following goals. First, we want an architecture to facilitate multilingual retrieval, minimizing the cost of adding a new language to the system. Second, we want to build a system that is not tied to any particular database and that in fact could be used for either single or multiple distributed databases. Third, we would like to maximize the usefulness/helpfulness of the system for domain experts who are not familiar with the structure of the database. For that set of users, who are the majority, the problem is that they frequently pose grammatical queries which are semantically valid but which are incomplete or ambiguous with regard to the way the information was encoded in the database.

The first of the goals is clear: we would like to be able to retrieve information in any language for which the system has a grammar, and we would like to minimize the cost of adding new grammars to the system.

The second of our goals is an issue of portability of the system. We would like to design a system that is database(s) independent (i.e., that is not tied to any particular database), avoiding the pitfall of earlier NL database systems that used semantic grammars tied to particular domains which precluded any extensibility or portability.

The third of our goals is motivated by the ambiguity that occurs in natural language. Elaborating on [Copenstake and Sparck Jones, 1990] we distinguish between two kinds of linguistic ambiguity: structural and phrasal ambiguity. An example of structural ambiguity can be found in the sentence: «Find the IBM computers and printers that are broken», which has at least the following possible interpretations: Find the computers made by IBM and the printers (made by any manufacturer) that are broken; find the computers and printers (all made by IBM) which are broken.

What we mean by phrasal ambiguity is that a lexical item or a phrase is ambiguous. The phrase can be ambiguous independent of the context where it appears or only with respect to the database. An example of the former (taken from [Copenstake and Sparck Jones, 1990]) would be «course» as in «How many courses are there?» where course could mean classes, golf courses, navigational courses or courses of a meal. Simple NL systems that have to deal with single small databases resolve this kind of ambiguity by having only one meaning (the only one relevant for the database) defined in the lexicon; but that is insufficient for multiple heterogeneous databases or even single large databases, where the different meanings do appear in the data. Phrasal ambiguity relative to the database refers to the case when a phrase or lexical item is incomplete or underspecified only with respect to a given database schema. The source of incompleteness or ambiguity can reside in the fact that the information encoded in the database is more specific than the query the user is posing to the system or in that the query as posed refers to multiple distinct properties or objects in the database. Let us present an example of this kind of ambiguity.

Assume the schema shown in Table 1 with three tables[1]. The first records density tests of materials that are molded, indicating time of measurement (i.e., before and after molding), and the method used to measure the density (i.e., gradient or null-pyco). The second table records the composition, name and some commercial properties of the material. The third table records the elongation and stress of the material at two points in time of a test: at the yield point (i.e., when the slope of the curve from the graph Force-Applied vs. Elongation-of-Material is 0) and at the break point (i.e., when the material breaks).

1: Density	2: Composition	3:Force/Elongation
Material Test-Number Gradient-Density-Before-Molding Gradient-Density-After-Molding Null-Pyco-Density-Before-Molding Null-Pyco-Density-After-Molding	Test-Number Primary-Element Secondary-Element Product Cost Hazardous-Properties	Test-Number Elongation-At-Break Elongation-At-Yield Stress-At-Break Stress-At-Yield

Table 1: A simple database schema.

In the query «which materials have density of 34?» the lexical item «density» has a non-unique interpretation with respect to the database. It could mean either of the four properties, all four of them, or a subset of them. Furthermore, it is only ambiguous with respect to this database schema. Had the schema been such that a single entry existed to record the density, irrespective of time and method of measurement, the query would not be incomplete.

[1] Traditionally, simpler schemas are used as examples in the literature but they are oversimplistic for the purpose of showing phrasal ambiguity and how to use a domain model to resolve it. For this reason we have taken this schema from a large existing chemistry database for the purpose of our examples.

This kind of incompleteness, which is more prevalent in complex domains where the names of fields and attributes are long descriptions that are meant to represent intricate relations or attributes of the entities recorded in the database, is what motivates our third goal. It falls within what has been referred in the literature as local pragmatics [Hobbs and Martin 1987] and is the sort of problem that, in order to be solved, requires a domain model that includes among other things information about the structure of the database schema.

In addition to a domain model, discourse context can also be used in resolving problems of local pragmatics (as well as pronoun resolution) as illustrated in the following example:

«Find the materials with break elongation of 15.35»
> list of materials
«what is their stress?»

Notice that in the second query (ignoring for the moment that we need the discourse context to resolve the referent for the pronoun «their») the word «stress» is again ambiguous with respect to the database schema which records the Stress-At-Break» and the Stress-At-Yield. We can resolve that ambiguity by exploiting the discourse context for the first sentence which contains an assertion about the elongation at break point. We can use the information about the break point in the previous sentence to make a guess that the user meant «stress at break» in the second sentence.

Hence, in order to achieve the goals of multilingual retrieval independent of a particular database and in order to address some of the issues that arise from ambiguity we want the system to produce a meaning-based representation of the queries to the system that is independent of the database. That meaning-based representation is produced by any general purpose grammar available to the system, along with the domain knowledge, thus making it database and language-specific independent. It constitutes the basis from which to exploit any domain and discourse model available to the system in order to resolve any phrasal ambiguity in the process of producing a database query.

C.- Using the same NL system for both applications

Having presented the goals for MT and database access, we make the observation that in both applications and in fact in many more applications involving a natural language interface, there is a need for both linguistic knowledge and a model of the domain. We will present some examples of each, and on the basis of that common knowledge needed and in the observation that both tasks use a meaning based representation of the input as an intermediate step, we will argue that it makes sense to reuse the same NL system for both applications.

In fact, both applications use a description of the grammar and the domain model to generate a semantically-based description of what was literally said. The grammar is used to generate the description where the terms are taken from the domain model. We make the initial observation that by having a single architecture, once a domain model or a grammar is developed, it can be readily used for either application.

1.- Linguistic Knowledge

The linguistic knowledge required for both tasks minimally includes a description of the grammar of the language in order to be able to do syntactic processing. We need to distinguish between pairs of sentences of the form shown in Figure 1 and Figure 2.

"which materials cost more than 30?"
"which materials cost no more than 30?"

Figure 1: Negation.

"which materials produced ethylene?"
"which materials are produce by ethylene?"

Figure 2: Active vs. Passive.

In addition to that, linguistic knowledge is required (also in both tasks) in order to handle pronoun resolution and to support a limited form of dialogue in database access. As shown below:

«which materials contain ethylene?»
> a list of materials
«which of them cost less than 30?»

Furthermore, all English queries shown in Figure 3 and all the Spanish ones shown in Figure 4 are equivalent with respect to the database. In order to interpret them as similar queries, we need to have linguistic theories of pragmatics that are applicable across different languages, such as Gricean principles [Grice, 1967]. Examples of such principles are to interpret declarative statements such as those in figures 3.C and 4.C as requests to find the objects that meet the description. Similarly, for figures 3.D and 4.D the most plausible interpretation in this context is that we want to return any materials with those properties as opposed to a yes/no answer; and that interpretation is equally applicable to both languages.

A.- "Find the materials with density before molding of 23"
B.- "What materials have density of 23 before molding?"
C.- "I wonder which molded materials have density of 23"
D.- "Is there any material that has been molded with density of 23?"

Figure 3: Equivalent set of queries in English.

A.- "Encuentra los materiales con densidad despues de moldearse de 23"
B.- "¿Cuáles materiales moldeados tienen densidad de 23?"
C.- "Me pregunto si hay materiales ya moldeados con densidad de 23"
D.- "¿Hay algún material con densidad de 23 despues de ser moldeado?"

Figure 4: Equivalent set of queries in Spanish.

Examples of additional linguistic knowledge required in MT to generate the more natural translation includes a description of the features that are marked or forced in the language. We use the

[2]This markedness distinction is similar to the one used in morphology [Jacobson, 1966], and is also related to the concept of blocking [Andrews, 1990].

concept of *markedness*[2] for pairs of similar lexical items that are discriminated from each other on the basis of a distinguishing property. The marked term is used whenever the distinguishing property is present. Otherwise, the unmarked term is used as the default. Examples of the forced features would be: nouns marked for number in English; nouns marked for number and gender in Spanish; levels of politeness in Japanese, etc. In Figure 5, the noun number which is forced in English for «boys» would be discarded when translating into Japanese, which does not enforce such a feature, rendering «Otokonoko». Another example is «pescado» which has to be marked in the lexicon as preferred, compared to the more general word «pez».

The boys	ran down	the street
Otokonoko-ga	toori-o	hashitteitta
Boy(s)-nom	street(s)-acc	ran down

Figure 5: Discarding information during translation.

2.- Domain Model

The need for a rich domain model independent of the database has been made in the literature as a reflection of earlier attempts to build interfaces for databases [Copenstake and Sparck Jones, 1989]. The first argument for it is that the data models used for database schema have clear and well defined semantics which do not correspond directly to natural language. Resolving the issues of ambiguity mentioned above can be regarded as involving specialized inference on the domain model. An example of a possible domain model needed to resolve the ambiguity in «which materials have density of 34?» is shown[3] in Figure 6. In the domain model we represent concepts that define general properties in the database such as density, elongation, etc. We also represent assertions about them that specialize those properties across different dimensions such as methods of measurement, temporal events involved in the measure, etc. Furthermore, we represent those specialized generic properties across the different dimensions described in the system. For example in Figure 6, we represent a concept for Density-Before-Molding and one for Density-After-Molding which contain assertions about temporal relations of when the density is measured (before and after molding respectively); similarly we encode Null-Pyco-Density and Gradient-Density which contain assertions about the method used to perform the measurement of the density (Null-Pyco for the former, and Gradient for the latter). Finally, we also represent the cross product of those concepts; and the result contains assertions about both temporal information as well as measurement methods: Null-Pyco-Density-Before-Molding, Null-Pyco-Density-After-Molding, Gradient-Density-Before-Molding and Gradient-Density-After-Molding. In the next section, we show how the information encoded in the domain model is used.

A second reason for having a domain model when using an NL system for database access, is to be able to answer meta level questions about the structure of the schema. Examples of such questions that would be enabled using a domain model would be the following: «What measures are taken when molding a material?», «When is the density of a material measured?», or «What are the methods for measuring density?», etc.

We now explain how the domain model can be used in MT to produce a natural translation. In the case of database access, the system maps from a representation in a rich domain model to a representation in a data model that has clear and well defined semantics. For machine translation the mapping is more complicated because the target is an equivalent representation of the source that is also drawn from the rich domain model, which does not

[3].- Only a small portion of the domain model is shown.

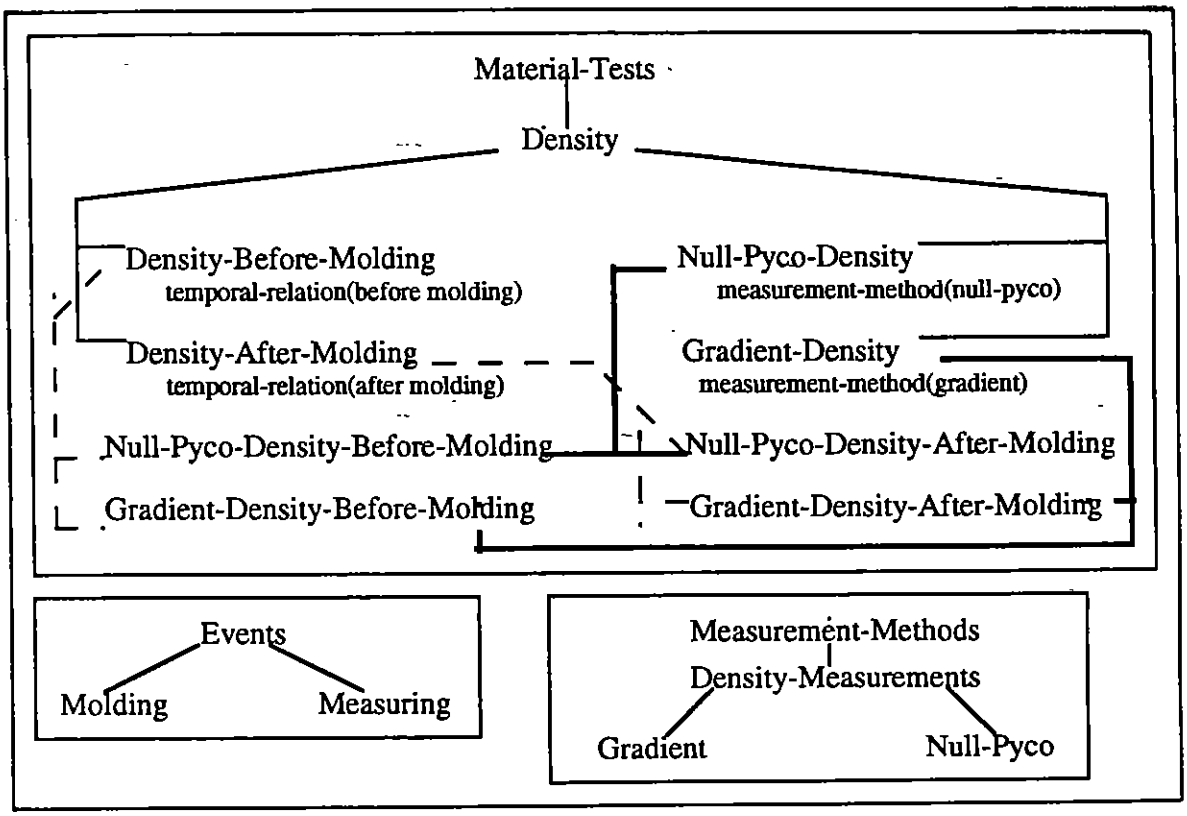

Figure 6: Fragments of a Domain Model.

have clear and well specified semantics. We need a rich model in order to have alternate ways of representing sentences, but the richer the model the more complex the mapping becomes. We use the domain model to get semantically-close words that are then discriminated on the basis of forced features in the target language, thus enabling us to consider «pez» and «pescado» when translating «fish» into Spanish, and selecting the latter if it can be inferred that the fish were caught or fit for food. We make the observation that this process is only local and that it is an active area of research [Kay and Peters, 1992] to define the sources of information required in a domain model in order to handle translation divergences in general. An example is the translation (taken from [Dorr, 1990]) of «John usually goes home» into Spanish «Juan suele ir a casa» (literally «John tends to go home»), where the emphasis of the sentence differs. In English, the main event is a «going» whereas in Spanish is a «tending».

II.- The structure of a portable NL system

A.- A linguistic representation of meaning

As we have seen, we use a linguistic representation of meaning as the basis for both applications. For MT, in order to support non-information-preserving translation [Barnett et al., 1991b] we use a deep, semantically-based interlingua (similar to systems like KBMT [Goodman, 1989]). In addition, we need a linguistic representation of what was literally said, including the notion of features that languages force speakers to make which differs from language to language. This is captured at the DRS level which is based on the Discourse Representation Structured as described in [Kamp, 1984, Heim, 1982]. In the present system both the interlingua and the DRS contain semantically-based and linguistic predicates; in fact as we will see the interlingua is equivalent to the input DRS.

Similarly for databases, we use the DRS to capture the literal meaning of what the user said. From the DRS, we start doing inferencing to figure out an interpretation of the literal meaning with respect to the database application (i.e., we try to map the literal meaning into one of the commands valid for database operations). It is similar to systems like TEAM [Grosz et al., 1987], LOKI [Binot et al., 1988] JANUS [Weischedel, 1989], and Natural Language [Ginsparg, 1983], in that it produces an intermediate meaning-based description of the input string, that is then translated into a database operator. But it is different from them in that the internal representation is not confined to be interpreted as a command to a database.

Figure 7 shows a schematic description of the system. The input and output strings are sentences from any language for which the system has a grammar. Words are defined in terms of objects in the domain model. The DRS, query language expression and target DRS are defined in terms from the model of the domain[4]. Besides providing a common ground for words to be defined, the domain model is used to infer meanings from the input. Thus, in the context of MT, that inferencing is the mechanism by which additional information that was omitted from the input string gets added. In the context of database interfaces, the inferencing is used to interpret the input string as a command to a database and to resolve problems of ambiguity.

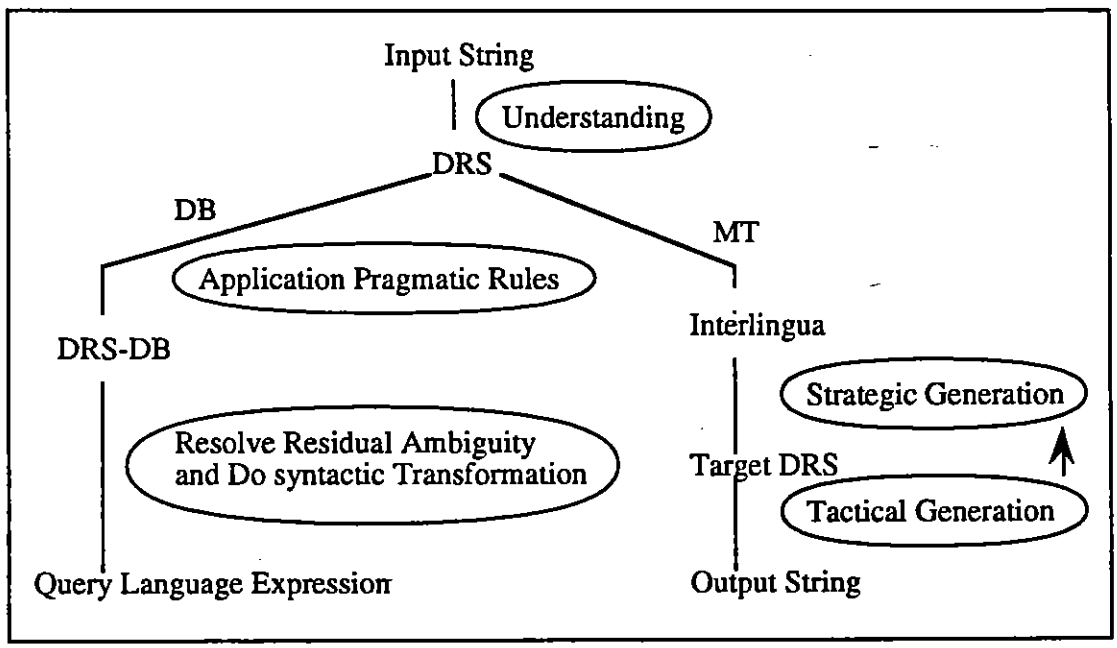

Figure 7: A General Architecture for MT and Database Access

The first thing that the system does, irrespective of the application, is to produce an internal representation of the input, the DRS. As mentioned before not all its terms are taken from the domain model, there are constructs that are fairly easy to deal with at a linguistic level but that become more complex (perhaps intractable) to deal with at a domain model level that supports inference. Examples of such constructs are quantifiers like «few». Also temporal relations are described with respect to speech time and reference time instead of an absolute time frame. Figure 8 (taken from [Barnett et al.,1991b]) shows a DRS for «There were five fish on the plate», where the marker *F indicates that its assertion was a feature forced in the input language (i.e., English in this case). Figure 9 shows the

[4] We assume that the domain-based representations can be treated as a set of logical assertions.

DRS for the query «find materials with density before molding of 23» and Figure 10 for «which materials have density before molding of 23?».

X: (Fish x) (measure x NaturalUnit 5)
Y: [(Plate y) (*F measure y NaturalUnit 1)] (Equal y previous-referent2)
Z: [(State z) (*F precedes z Now)] (supported-by z x y)

Figure 8: A DRS for «There were five fish on the table».

X: (finding x)
Y: (material y) (found-object x y) (measure y plural)
Z: (z density y v)
W: (molding w) (temporal-relation before z w)
V: (Equal v 23)

Figure 9: A DRS for «Find materials with density before molding of 23».

X: (material x) (measure x plural)
Y: (y wh-operator x)
Z: (z density y v)
W: (molding w) (temporal-relation before z w)
V: (Equal v 23)

Figure 10: A DRS for «Which materials have density before molding of 23?».

In the next section we'll describe what happens after the system produces a DRS; this depends on the specific application that the system is being used for.

B.- Interpreting the DRS in the context of each application

1.- MT

In the case of MT, once a DRS has been produced the next step is to map it to an interlingua expression. This process at the moment is the identity function given that we have not found a need to add any information at this stage. From the interlingua representation, we want to obtain a DRS for the target language. It is at this point that information about what to say in the target text is decided. Using the information in the domain model as well as the features that are forced and unforced in the source and target language, assertions can be added or discarded from the interlingua expression in producing a target DRS. The target DRS should be a structure more suitable for generating natural sounding text in the target language. The last step is to produce a target string from the resulting DRS. These last two steps (referred to as the strategic and tactical components of a generation system) are often times interrelated. For a more detailed description of the MT architecture, see [Barnett et al., 1991a] [Barnett et al., 1991b] [Barnett, Mani, Rich, 1991].

2.- Database Interfaces

In the case of database interfaces, once a DRS has been produced it needs to be interpreted as an operation to the database, or about the database as in metalevel questions about the structure of the global schema. Two levels of pragmatic rules are applied to the DRS to produce an operation for the

database. The first is a set of generic application pragmatic rules that are applicable across different domain models. Many of them are also applicable across different languages. The second level is a set of specific application pragmatic rules that exploit the domain model and are mainly concerned with mapping predicates from the domain model into those defined for the global schema of the database. Once those pragmatic rules have applied, a third step is to resolve any residual ambiguity present in the query, as well as to do some simple syntactic transformations to conform to the syntax of the query language used by the back-end database.

a.- Generic Application Pragmatics

A set of generic application pragmatic rules that are applicable across all domains and often across languages apply first to the DRS. An example of such a rule is to interpret declarative statements where the main verb refers to «ponder» or «wonder» event in the domain model as a request (as in the examples shown in figures 3.C and 4.C). Another example would be to interpret yes/no questions where the main verb refers to the «existence» of an object, to return a list of all such objects instead of just returning a yes/no answer (as in the examples of figures 3.D and 4.D). Yet another example of such transformation would be for imperative transitive statements where the main verb connotes a «find» or «search» event, to translate the assertion about the main verb[5] as a request and identify its direct object as the main focus of the query (as in the examples from figures 3.A and 4.A). Figure 11 shows the result of these transformations to the DRS for «find materials with density before molding of 23» that was shown in figure 9. In fact, we currently are discarding any modifiers to the top verb as in «Quickly find all objects with x properties»; where the modifiers could indicate useful information on how to conduct the search[6] or on how to respond in a cooperative manner [Kaplan, 1982].

```
Y: (material y) (measure y plural) (focus y)
Z: (z density y v)
W: (molding w) (temporal-relation before z w)
V: (Equal v 23)
```

Figure 11: Result of applying application pragmatic rules to DRS in figure 9.

b.- Specific Application Pragmatics

The second level is a set of specific application pragmatic rules which exploit the domain model and which are mainly concerned with the task of mapping generic predicates from the domain model into those defined for the global schema of the database (which are also defined in the domain model but are only a small subset of it). Notice that these rules are applicable for either a single database or multiple distributed heterogeneous databases integrated into a global schema that encompasses a global view of all the individual pieces as described in [Buneman et al., 1990], [Collet, Huhns and Shen, 1991]. In fact, as long as the resources are integrated using a rich domain model, it is transparent to the natural language system whether the back-end is a single database or multiple heterogeneous database systems. The only information that the NL system has access to is a global schema and it is the goal of the NL system to map natural language expressions into expressions defined in terms of that global schema, and special predicates such as quantifiers, logical operators, etc. (We assume that the NL

[5].- Notice that this transformation applies only at the top level since similar verbs could be embedded in the input as in «Find all dogs used to find people trapped in the snow».

[6].- More care has to be applied when performing this transformation; it is part of our future work.

system does no query query optimization). It is then the responsibility of the back-end database manager to distribute the queries accordingly to the appropriate databases and to optimize each query[7].

These operations require that certain information be recorded in the domain model. In particular we need to know which of the predicates in the domain constitute the global schema. From Figure 6, the predicates Null-Pyco-Density-Before-Molding, Null-Pyco-Density-After-Molding, Gradient-Density-Before-Molding and Gradient-Density-After-Molding are part of the global schema.

The task of mapping from generic predicates from the domain model into those defined for the global schema of the database is essentially a classification operation (also referred in the literature as term subsumption). Term subsumption languages [Patel-Schneider, et al., 1990] are representation formalisms that specify formal semantics for the definition of terms and that deduce whether a term along with its assertions is subsumed by (i.e., is more specific than) another term. Given an input set of terms constituting a description of an entity, the term subsumption language should find the most specific term matching that description. From a classification viewpoint, given the set of input terms reified as a class, the classification should find the most specific one matching the input class. Let us take as an example the application of the rules to the DRS in Figure 11. The result of applying the rules to the term Density along with any assertions made about it (the temporal relation in Z) should return a more specific term in the domain model that contains those assertions as the result (namely Density-Before-Molding). The result is shown in Figure 12; where the predicate Density-Before-Molding is defined in the domain model as having an assertion about temporal-relation (Before Molding).

```
Y: (material y)
Z: (z density-before-molding y v)
V: (Equal v 23)
```

Figure 12: Result of applying specific application pragmatic rules to DRS in figure 11.

c.- Resolving residual ambiguity with respect to the database

The next step in producing a query from the DRS is to validate the predicates against the database schema. If a predicate is not defined to be part of the global schema the input was ambiguous or incomplete with respect to the database. At that point the system uses a decision tree search procedure utilizing the information in the domain model to find an unambiguous predicate(s) defined in the global schema. At each point in the decision tree the system uses the information present in the domain model that records what information is required to further specify a predicate and prompts the user for such information (the implementation details on how this process works are outside the scope of this paper). For example in the case of the DRS in Figure 12, the term Material is part of the global schema, and the term Equal is a valid operator. The term Density-Before-Molding however is not complete with respect to the global schema; the two terms subsumed by Density-Before-Molding which are defined in the global schema are Null-Pyco-Density-Before-Molding and Gradient-Density-Before-Molding. The information required to know which one of the two was meant is what type of measurement method was used. (This is taken from the domain model). At this point the system presents the user with a list of options specifying the possible measurement method used for density and would ask the user to select one of them. An example of such dialogue is shown in Figure 13. Once the user selects an option all of the terms in the DRS will have a unique, non-ambiguous interpretation with respect to the

[7].- This assumption that we first produce a query from the DRS which is later optimized may prove inadequate, in which case a tighter coupling between the two processes might be necessary.

database. The only remaining step is to do a simple syntactic transformation to conform to the syntax of the database query language used by the back end database.

What measurement method was used:
1.- Gradient
2.- Null-Pyco
3.- Either one

Figure 13: A Sample Disambiguation Dialogue.

III.- System Status

We have implemented a system with this architecture, the KBNL system [Barnett et al., 1991a]. The KBNL has three main components: an understanding system, Lucy [Wittenburg and Barnett, 1988], [Rich and Luperfoy, 1988], which performs the mapping from input to DRS; secondly a generation system, Koko [Barnett and Mani, 1990] which is used to perform the interlingua to string mapping, and thirdly, a lexical acquisition tool, Luke [Wroblewski and Rich, 1988], which assists in building the lexicon and mappings to the domain model. All of these systems exploit a generic interface to representation systems used to encode the domain model [Barnett, Martinez, Rich, 1991]. This generic interface enables the system to be portable across different platforms that conform to the specification. There currently is an English and a Spanish grammar for the system. A set of bidirectional semantic rules has been implemented to handle noun compounding and cases of metonymy [Barnett and Mani, 1990]. With respect to the database interface, there is a small fourth module which currently does the query validation, and syntactic transformation to produce query language expressions. A large domain model in the area of chemistry has been defined for an application involving large heterogeneous databases.

IV.- Conclusions and Future Work

We have presented a multilingual NL system architecture that can be used for both database independent retrieval and machine translation. There are still many issues in each of the two applications to be resolved. For MT, one of the main problems is to identify what sources of information need to be encoded in the domain model to handle non-local translation mismatches. For more details on future work on MT see [Barnett, Mani, Rich, 1991]. For database access, some of the issues for future work are how to integrate large-scale shared domain-models, whether the NL systems needs to optimize the queries it produces or if that should be left entirely to the back-end database system. Other issues are the distributivity and reversibility of conjunctions and disjunctions; and the robustness and coverage of the system. For a good discussion on requirements of an NL system interface for databases, see [Templeton and Burger, 1986].

V.- Acknowledgements

I would like to thank the following members of the KBNL project: Jim Barnett, Zuzana Dobes, Inderjeet Mani, and Elaine Rich for their comments and contributions to this paper.

References

ANDREWS, A. «Unification and Morphological Blocking», Natural Language and Linguistic Theory, 8(4), November 1990, pp 507-558

BARNETT, J., D. D'SOUZA, K. KNIGHT, I. MANI, P. MARTIN, E. RICH, C. AONE, J. BLEVINS, W. BOHRER, S. LUPER-FOY, J.C. MARTINEZ, and D. NEWMAN, «Knowledge-Based Natural Language Processing: the KBNL System», MCC Technical Report ACT-NL-123-91, 1991

BARNETT, J., MANI, I., RICH, E., AONE, C., KNIGHT, K., MARTINEZ, J.C., «Capturing Language-Specific Semantic Distinctions in Interlingua-Based MT», Proceedings of Machine Translation Summit III, Washington, D.C., July 2-4, 1991.

BARNETT, J., MARTINEZ, J.C., RICH, E. «A Functional Interface to a Knowledge Base: an NLP Perspective.» MCC Technical Report ACT-NL-393-91, 1991.

BARNETT, J., MANI, I., and RICH, E. «Reversible Machine Translation: What to do when the languages don't match up», Proc. of the ACL Workshop in Reversible Grammars in NLP, Berkeley June 1991.

BARNETT, J., K. KNIGHT, I. MANI, and E. RICH, «Knowledge and Natural Language Processing», Communications of the ACM, August, 1990.

BARNETT, J., MANI, I., RICH, E., AONE, C., KNIGHT, K., and MARTINEZ, J.C., «Knowledge Based Machine Translation» in «Frontiers in Knowledge-Based Computing», Narosa, New Delhi. 1990.

BARNETT J., and I. MANI, «Using Bidirectional Semantic Rules for Generation», Proceedings of the Fifth International Workshop on Natural Language Generation, pp 47-53, Pa., 3-6 June, 1990.

BINOT, JL, et al., «LOKI: A Logic Oriented Approach to Data and Knowledge Bases Supporting Natural Language Interaction», London, 1988, Scicon Ltd.

BOBROW, JR, RESNIK, P., AND WEISCHEDEL, R., «Multiple Underlying Systems: Translating User Requests into Programs to Produce Answers», In Proc. the 28th Annual Meeting of the ACL, 1990.

BUNEMAN, O.P., DAVIDSON, S.B., and WATTERS, A., «Querying Independent Databases», Information Sciences, Vol. 52, Dec. 1990, pp 1-34.

COLLET, C., HUHNS, M., and SHEN, W. «Resource Integration using a Large Knowledge Base in Carnot», Computer, Vol. 24, No. 12, Dec. 1991, pp 55-62.

COPENSTAKE, A and SPARCK JONES, K. «Natural Language Interfaces to Databases», Knowledge Engineering Review, vol. 5, no. 4, Dec. 1990, pp 225-49.

COPENSTAKE, A and SPARCK JONES, K. Inference in Natural Language Front Ends to Databases. Technical Rport 1963. University of Cambridge: Computer Laboratory. 1989.

DORR, B. «Solving Thematic Divergences in Machine Translation», Proceedings ofthe 28th Annual Meeting of the ACL, Pittsburgh, 1990.

GRICE, P., «Logic and Conversation» in William James Lectures, Harvard University, 1967.

GROSZ, B., ET AL., «TEAM: an experiment in the design of portable natural-language interfaces», Artificial Intelligence 12, 1987, pp 173-243.

HEIM, I. «The Semantics of Definite and Indefinite Noun Phrases», University of Masachusetts, Ph.D. Dissertation, 1982.

HOBBS, J. and MARTIN, P. «Local Pragmatics», Proceedings of the Tenth International Joint Conference on AI, Milan, Aug. 1987.

JAKOBSON, R., «Zur Struktur des Russischen Verbums», in Hamp, Householder, and Austerlitz, eds. Readings in Linguistics, II, Chicago: The University of Chicago Press, 1966.

KAMMEYAMA, H., OCHITANI, R. AND S. PETERS, «Resolving Translation Mismatches with Information Flow», Proceedings of the 29th Annual Meeting of the ACL, Berkeley, 1991.

KAMP, H. «A Theory of Truth and Semantic Representation», in M. Groenedijk, J. Janssen, and M. Stokhoff, eds., Formal Methods in the Study of Language, Dordrecht: Foris, 1984.

KAPLAN, SJ., «Cooperative responses from a portable natural language query system», Artificial Intelligence, 19, 1982, pp 165-187.

KAY, M., and PETERS, S., «Machine Translation by Negotiation», Unpublished, 1992.

PATEL-SCHNEIDER, P. , et al. «Term Subsumption Languages in Knowledge Representation», AI Magazine, Summer 1990.

RICH, E. and S. LUPERFOY, «An Architecture for Anaphora Resolution», Proceedings of teh Second Conference on Applied Natural Language Processing, Austin, 9-12 February, 1988.

TEMPLETON, M., BURGER, J. «Considerations for the Development of Natural-Language Interfaces to Database Management Systems», Cooperative Interfaces to Information Systems. Springer, 1986.

WEISCHEDEL, RM, 1989. «A hybrid approach to representation in the JANUS natural language processor», Proceedings of the 27th ACL, Vancouver, British Columbia, 1989, pp 193-202.

WITTENBURG, K., and J. BARNETT «Canonical Rpresentation in NLP System Design: A Critical Evaluation», Proc. of the Second ACL Conference on Applied Natural Language Processing, 1988.

WROBLEWSKI, D., and E. RICH, «LUKE: An Experiment in the Early Integration of Natural Language Processing», Proc. of the Second Conference on Applied N. Language Processing, 1988.