A SEMANTIC DATA MODEL FOR INTELLECTUAL DATABASE ACCESS

Toyohide Watanabe
Yuusuke Uehara
Yuuji Yoshida
Teruo Fukumura

Reprinted from PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON COMPUTERS AND COMMUNICATIONS,
Scottsdale, AZ, March 21-23, 1990

A Semantic Data Model for Intellectual Database Access

Toyohide WATANABE, Yuusuke UEHARA, Yuuji YOSHIDA and Teruo FUKUMURA

Department of Information Engineering
Faculty of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-01, JAPAN

Chukyo University
101 Tokotate, Kaizu-cho,
Toyota 470-03, JAPAN

***** Abstract *****

The traditional data models provide basic frameworks for database structures. However, they do not include semantic information for the representation and manipulation of databases. The data model is a representation medium concerning computer phenomena, but is not an abstraction mechanism with respect to human activities. Therefore, it is necessary to manage semantic information with a view to the intellectual information retrievals and effective database managements.

In this paper, we address a semantic data model to make the intellectual database access possible. Our semantic data model is fundamentally designed on the basis of the ER model from a viewpoint of the management of semantic information. Moreover, the concept in the case grammar is introduced not only to make the properties of entities distinct, but also to interpret natural-language-like queries effectually.

***** Keywords *****

semantic data model, ER model, case grammar, intellectual database access, entity, relationship, attribute, object, class, semantic database schema, matching mechanism

1. INTRODUCTION

The traditional data models used in the hierarchical, network and relational databases do not preserve semantic information for database organizations and mutual relationships among the composite entities, though they can control databases systematically, concerning effectual database managements.[1] Many currently employed database management systems force users to grasp semantic information about databases in advance, in addition to the operational usage of system functions. Such database handling facilities are neither effectual nor successful for end users. As one of simple solvable methods for this issue, the menu-driven interaction technique has been applied to peculiar database environments. Although this technique is powerful enough to control users' requests in application-specific predetermined processes, the adaptable ranges and the data manipulation abilities are very limited.

On the other hand, the subject for intellectual database accesses has been studied as one practical application of natural language processing techniques. In this approach, the natural language processing technique is applied as the query translation mechanism from natural-language-like queries to the corresponding database manipulation procedures.[2] However, this natural language processing module is too strongly dependent on an application-specific database because the module is developed only on the basis of particular content information about the database. This approach is short of the generality though users' manipulation interfaces become more natural in comparison with the traditional manipulation languages such as SQL. This is because in this approach a problem of the semantic gap between data models and the real world is not studied at all, but the research objectives focus on developing effectual query forms for the database retrievals.[2]

In this paper, we propose a semantic data model to address the issue of the semantic gap directly, concerning the subject for intellectual database accesses. In particular, our semantic data model is designed so that database management systems, by themselves, can manage autonomously structural and semantic information about database organizations, relationships among the composite entities and so on. A fundamental framework in our semantic data model has not only the features of the ER model, but also introduces the concept of the case grammar. Our model is composed of entity sets and their mutual relationships as well as the ER model. However, we do not distinguish the attended attributes from entities: that is, every object is the entity. Such a framework is similar to that in the case grammar. Namely, our semantic data model is adaptable to control the database schema and successful to interpret natural-language-like queries. Additionally, our model assigns the relation roles to individual entities, specified by appropriate relationships. This concept is introduced so as to make characteristics of the mutually related entities to be defined in databases clear: individual entities have always the inherently assigned roles in accordance with main events such as verbs.

2. SEMANTIC INFORMATION IN DATABASE

The database is not a single collection of various types of data, but must represent a part of all the facts that could be extracted from the real world at a time. In the traditional database management systems the database definition language specifies only structural information about the constructive organization of collected data.

The semantic information, concerned with domain constraints for data items, mutual relationships among them, functional properties of data item classes and so on, is understood only by database designers, but can not be defined explicitly in databases. The database management system manages the composite data through the database schema, which controls mainly the corresponding constructive information between the logical and physical structures. This mechanism forces users to understand every requisite information about individual data items of the database structure, various relationships among the composite data, effectual domains for individual data items and so on. Without such knowledges, users can not manipulate databases effectively. In the database management systems based only on the traditional data models, the definition and manipulation facilities for databases are very limited because the data models are partial abstractions of computer phenomena involving files and computer processings.[1]

The semantic data model provide definition and manipulation abilities of such semantic information, in addition to the conventional structural information.[4] The database management systems based on such semantic data models can accept more ambiguous and flexible queries(e.g. natural-language-like queries) than the traditional frameworks. The semantic data models offer abstractions of the real world because they can capture the meanings of information and its behavior with regard to human activities pertaining to the application.[1] Therefore, the semantic data models must represent the facts or phenomena observed in our real world: the properties of entities, the relationships among entities, the properties of relationships, the relationships between relationships and entities, and so on. Until today, many semantic data models such as ERM, SDM, FDM and IFO[4, 8] have been proposed progressively in order to set about the above issues. However, many of them do not always provide sufficient frameworks to establish the databases with intellectual manipulation interfaces.[5, 6]

3. A SEMANTIC DATA MODEL

An ER(entity-relationship) model was firstly proposed as one of the semantic data models.[3] The principal concepts are entities and relationships. It is very convenient to capture the real world state by the entities which we can distinctly identify, and relationships which represent the mutual associations among entities. In many other semantic data models,[4, 8] the similar interpretation is applied: objects, object classes and relationships Our semantic data model is fundamentally based on this interpretation with a view to capturing the real world state.

3.1 FRAMEWORK BASED ON ER MODEL AND CASE GRAMMAR

Our basic concepts to deal with the semantic information are objects, classes and relations: a class corresponds to the object class(or the entity set); and a relation does to the relationship in the ER model. Our objects are divided into primitive objects and compound ones in point of the structural organization: the compound object is composed of the primitive objects and/or compound ones by the aggregation operation. Additionally, the objects are classified to concrete objects and abstract ones from a viewpoint of operational properties. The concrete objects can be distinguished in themselves as the existing entities, while the abstract ones are conceptual identifiers. The existing entities must be specified with each other through the abstract objects. For example, a man called by the name "A", is distinguished from another man of the name "B" under the interpretation that the abstract class is the "name" and the concrete class is the "man". Here, "A" and "B" are values belonged to the "name". In many cases, we can recognize and distinguish the existing entities through the abstract objects.

Table 1 Kinds of roles

role	meaning	role	meaning
agt	agent	obj	object
at	location	time	time
recp	recipient	inst	instrument
from	source	to	goal

The classes are the object sets and are also divided into individual categories, corresponding to the object concept. Here, we introduce the concepts of the pointer class and the key class. The pointer class is an abstract class to identify the concrete class as a set of data values: for example, the "name" is this pointer class. In another word, the pointer class corresponds to the attribute domain of candidate keys in the traditional databases. On the other hand, the key class is identical to the attribute domain of primary keys and can distinguish concrete objects individually under the 1-1 correspondence mapping.

The relation is a relationship to be defined among classes as well as the relationship in the ER model. In this case, the roles which indicate the meanings for relations are assigned to the related classes. The concept of roles is the same as the case concept in the case grammar. The case grammar, proposed by C.Fillmore, is useful to interpret the natural languages.[7] The basic framework is to construct the conceptual dependency structure by looking upon a verb, which composes a sentence in the natural language, as the central organization unit and then by assigning the term attributes(cases) to the other words, respectively. Therefore, the ER model, whose mechanism assigns relationships to verb phrases and entities to noun phrases, is adjustable to the basic framework of the case grammar concerned with the correspondence of each concept. Namely, our roles are very similar to the cases with respect to their conceptual effects. The kinds of our cases, as shown in Table 1, are fundamentally derived from Fillmore's cases.

With these concepts, we manage the semantic information about databases. Now, we will illustrate our semantic data model concretely by using our diagram based on the ER notation: the classes are depicted by rectangular boxies; and the relations are by diamonds. The diagram in Fig.1

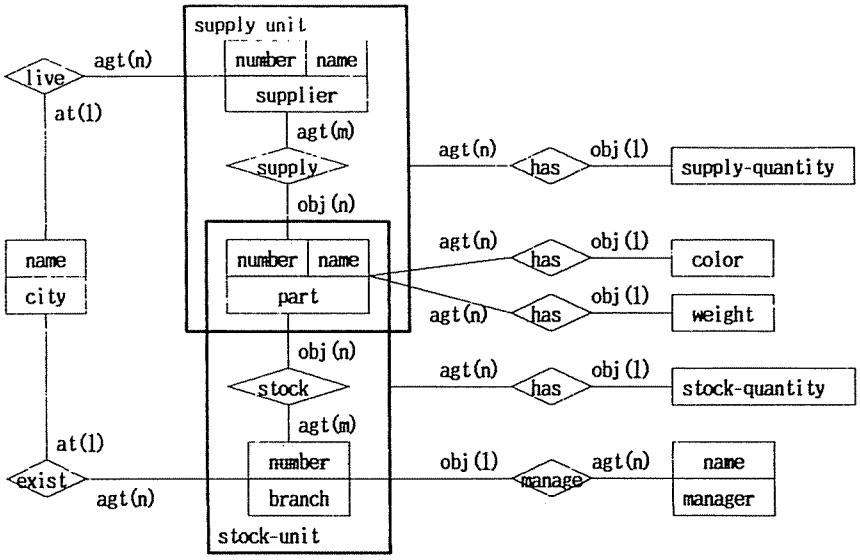


Fig.1 Representation in our semantic data model

represents the following facts:

" The suppliers, who live in cities, supply parts by the quantity. Each branch, organized by the manager, accommodates several kinds of parts. Individual suppliers are distinguished by the unique identification numbers and names. The branches are identified by the branch numbers, and located to cities. The parts are specified by the unique numbers and part names, and characterized by their colors and weights."

In the relational databases, we can manage the similar data files by using a set of several tables as shown in Fig.2. In comparison with the relational data model, individual entities to be defined in our model are distinguished clearly in accordance with their mutual relationships and roles. In Fig.1, the key classes are indicated by hatched boxies, and assigned to the corresponding concrete classes in order to indicate the concrete ones uniquely one by one. Also, the pointer classes are assigned to the concrete classes in order to distinguish each object. Moreover, the roles such as "agt", "obj" and "at", and cardinalities such as "1:1", "1:n" and "n:m" are specified together on the lines depicted between the classes and relations. Additionally, the compound classes such as "supply-unit" and "stock-unit" are shown by bold-type rectangular boxies, which include the classes and the relations as elements. From this diagram we can observe that our semantic data model is not related to the ER model directly, but rather derived from the case grammar.

3.2 CHARACTERISTICS OF OUR SEMANTIC DATA MODEL

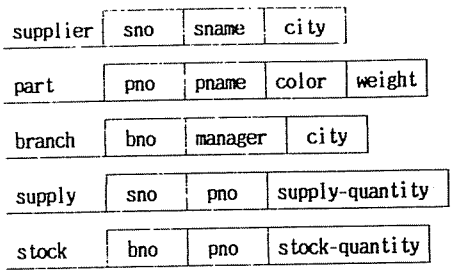
Our semantic data model is not only based on the framework of the ER model, but also characterized by the concept of the case grammar. The object classes represented in our model have individually distinct properties with respect to their mutual relationships. Therefore, our semantic data model is a more powerful semantic representation tool than the ER model.

Our semantic data model is different from the ER model in the next main features:

- exclusion of attributes,
- introduction of roles.

Before explaining these concepts, we represent the database schema illustrated in Fig.1 by the ER diagram. The diagram is shown in Fig.3. In the ER model, each entity is characterized by the definitely attended attributes, which assign the properties of entities to the value-sets. The attribute "city" is meaningful for the entities "supplier" and "branch" as the corresponding property. The attribute "city" in the entity "supplier" is different from that in the entity "branch" conceptually though they belong to the same object class. Of course, the "city" may be defined as the entity if the "city" must be composed of several other attributes such as the population, the area, the mayor and so on. However, in this case the "city" is not always manipulated as the entity because the "city" is not an interesting object.

The distinction between attributes and entities is relative and ambiguous. At a specific time "t", a database designer distinguishes attributes from entities after he has firstly selected the objects, which are mostly interesting for him, as the entities from his conceivable phenomena. However, another database designer may disagree with such a decision. Thus, it is different to distinguish entities and attributes clearly. In order to adjust the databases for a long time, the distinction between attributes and entities is injurious. Moreover, our semantic data model must be adaptable to the conceptual dependency structure for a natural-language-like query. It is required that our model is powerful to manage the databases and effective to interpret natural-language-like queries. In order to analyze the natural-language-like



(note) sno : supplier-number, pno : part-number,
bno : branch-number, sname: supplier-name,
pname: part-name

Fig.2 Representation in relational database

query effectively, it is not successful to distinguish entities and attributes because the query does not always specify individual terms and their relationships in the database structure, sufficiently.

Usually, the composite elements in sentences are words, but neither entities nor attributes. In case of interpreting the natural-language-like queries, we can distinguish neither entities nor attributes, defined in the databases, unless they are specified by the operational commands "SELECT", "FROM", "WHERE" and so on like SQL. Namely, our semantic data model must be designed so as to provide the context information, concerning database retrieval support, and also to make it possible to translate the ambiguous and information-less queries into the formal-terms commands. At least, the distinction between attributes and entities is not useful to interpret sentences. If the semantic data model was compatible to the conceptual structure of a grammar of some language without the conceptual distinction between attributes and entities, the natural-language-like queries can be analyzed easily. From such a consideration point of view, we excluded the concept of attributes from our semantic data model.

Moreover, we introduce the concept of roles in order to make mutual relationships among entities clear. The roles are assigned to individual entities defined through the relationships. Thus, it is successful to interpret the meanings of databases. For example, even if the natural-language-like queries were short of complete information for retrieval procedures, our semantic data model can compensate the deficient information by the matching mechanism between entities and relationships. At least, our roles, attended inherently to entities, take important roles to analyze ambiguous and incomplete queries so that the case grammar can represent conceptual dependency structures of natural languages successfully.

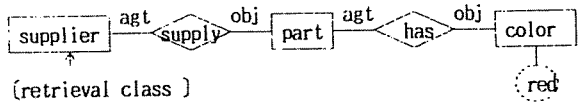


Fig.4 Internal query form

4. QUERY FORM

Natural-language-like queries are firstly translated into internal query forms, and then the internal query forms are executed by checking up with our semantic database schema. Namely, the internal query form is composed on the basis of the representation structure similar to the semantic data model. Also, though the semantic database schema is derived from our semantic data model, it includes more information: information concerning classes and relations, specified by the database definition language; and object information, gathered from databases.

For example, consider the next query: " Who supplies red parts ? ". This query is translated into the following command in SQL:

(ex.)

```
SELECT SUPPLIER.SNAME
  FROM SUPPLIER, SUPPLY, PART
  WHERE PART.COLOR='red' AND PART.PNO=SUPPLY.PNO
    AND SUPPLY.SNO=SUPPLIER.SNO
```

The SQL retrieval command must navigate among 3 tables in the database shown in Fig.2. Thus, it is impossible to translate the original query into the SQL command without any knowledge of the database schema. Users are required to be familiar with various kinds of knowledges in advance: the database structure, the table structure, the relationships among tables, and the relationships between retrieval conditions and reference items. In our framework, such a problem is resolved easily. The previous query is translated into the internal query form as shown in Fig.4. This form is similar to our semantic data model. Additionally, this form preserves the structural and semantic information of the original query. Therefore, it is not difficult to check whether the internal query form matches with a part of the semantic data model. For example, we illustrated the matching mechanism in Fig.5, for the internal query form in Fig.4 and the semantic data model in Fig.1. Thus, we can retrieve the values of suppliers' names through the semantic database schema.

Our internal query form and semantic database schema are internally represented by list structures. For example, the internal query form is as follows:

(ex.)

```
( SUPPLIER "?" ( SUPPLY agt ( PART null ( HAS agt
```

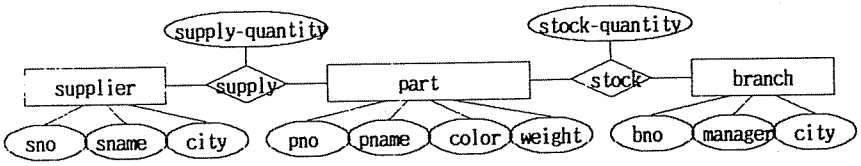


Fig.3 Representation in ER model

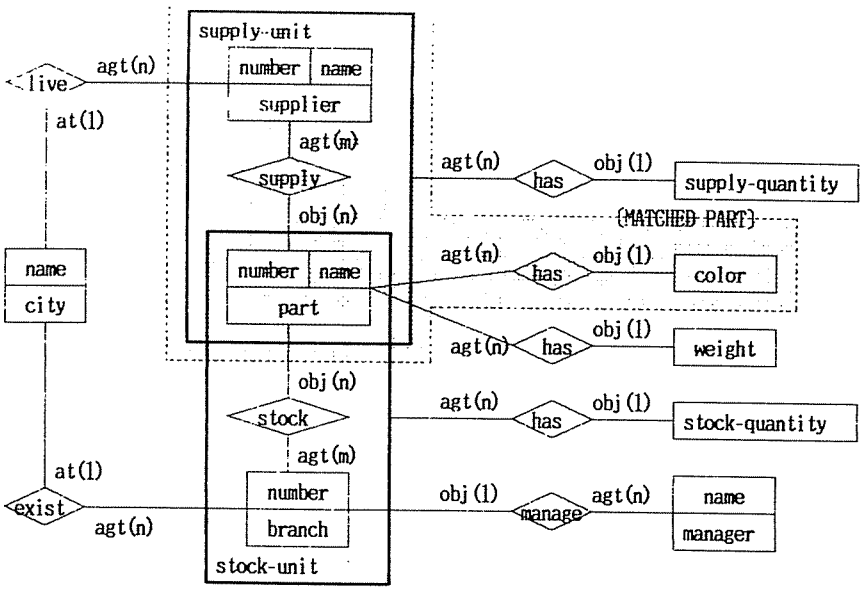


Fig.5 Matching mechanism between semantic data model and internal query form

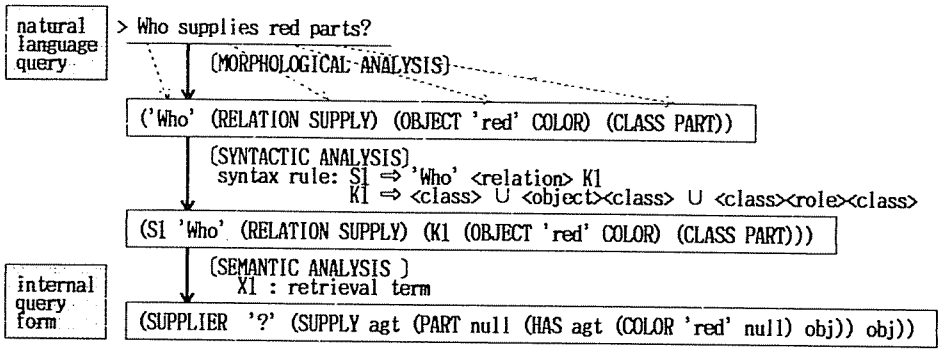


Fig.6 Translation process of query

(COLOR 'red' null) obj)) obj))

Namely, the basic frames are: as for the object class,

(class-name, value, relation-link);

and as for the relation,

(relation-name, role for relating-class,
related-class, role for related-class).

Here, the distinction between the related-class and relating -class is not clear, but they are selected arbitrarily from the classes linked to the relation. The symbol "?" in the object class frame represents that this class is the retrieval item.

In Fig.6, we show the translation process briefly. The translation from a natural-language-like query to our internal query form is mainly divided into 3 steps: the morphological analysis; the syntactic analysis; and the semantic analysis. The procedures in these steps refer to the semantic database schema in order to supplement the deficient information, if necessary. In the morphological analysis, the distinction about whether individual words are the relations, the classes or the objects is dependent on their registered information, specified by the database definition language. In the syntactic analysis, syntactic rules are predefined in advance. For example,

(ex.)

S1 ⇒ 'who' <relation> K1

K1 ⇒ <class> U <object> <class>
U <class> <role> <class>

are applicable to our example query as shown in Fig.6. Finally, in the semantic analysis, the translation is performed easily by using the class, object and relation knowledges, in addition to the reference of the semantic data model.

5. DISCUSSION

Our semantic data model manages semantic information about databases so that the database, by itself, is organized as a collection of autonomous objects. The basic idea in our model is derived from the framework of the ER model and the concept of the case grammar. This is because our model must be not only smart to manage databases

intellectually, but also applicable to interpret natural-language-like queries effectually. In order to analyze the natural language functionally, our model must distinguish individual terms clearer than the ER model as well as the semantic network. It is difficult to interpret the words of sentences by the ER model which separates entities and attributes, attached to the entities, structurally. Generally, the natural languages queries do not reflect structural and semantic information about the database organization in detail. Our semantic data model must provide an interpretation ability to be flexible for various types of queries, in addition to an adaptable representation ability of semantic information about database structures.

At present, the types of natural-language-like queries, that we are ready to manipulate databases, are as follows:

(1) retrieval of data instances:

(ex.)

Who supplies red parts ?

Namely, this type is composed of questional words such as "who", "whose", "whom", "what", "where", "when" and so on, as the starting word. Usually, the database retrieval commands belong to this type.

(2) question about facts:

(ex.)

Does Mr.Smith supply red parts ?

The answer is composed by "yes" or "no", as the starting word. Therefore, the evaluation of this type of query is firstly to retrieve data instances, and then to compare these instances with the key phrase of the query.

(3) selection of a data instance:

(ex.)

Which is higher, Mt.Fuji or Mt.Everest ?

The answer is to select one term from the terms, denoted in the query. In this case, the evaluation is firstly to retrieve data instances by looking up the terms as the retrieval conditions respectively, and then to compare their data instances by some measure.

Additionally, the aggregation queries are acceptable such as the summation, the average, the maximum, the minimum and so on. In these queries, the aggregation operators are applied to the data instances, which are retrieved in the queries of the type (1).

(ex.)

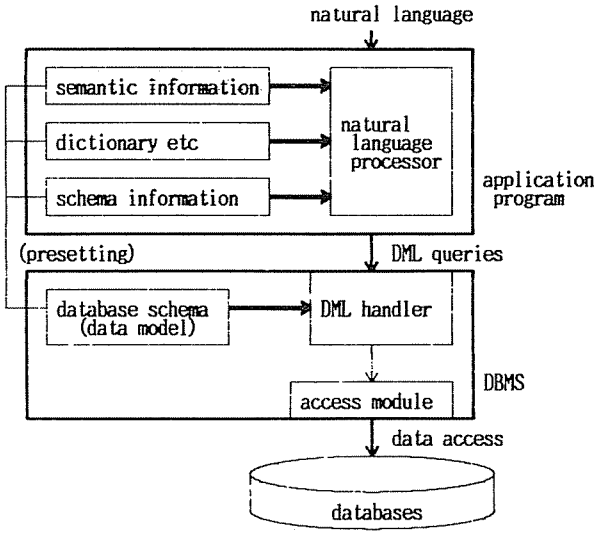
aggregation query : What is the average weight of parts ?

⇩(transformation)

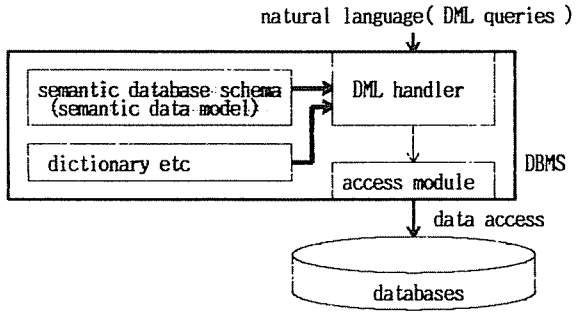
retrieval of data instances : What are the weights of parts ?

aggregation operation : (average operation)

Our subject for the intellectual database access is very similar to the objective of the traditional intellectual database researches in point of supporting the natural-language-like queries as database manipulation languages(DML). However, the database management systems(DBMS) derived from the adoption of semantic data models provide a more general framework,[5] in comparison with the traditional



(a) NL approach(as application)



(b) semantic approach(as database management system)

Fig.7 Interfaces for intellectual database accesses

approaches based on the applications of natural language processing techniques.[2] We call the intellectual database research based on the semantic data model as the semantic approach, and the intellectual database research based on the natural language processing as the NL approach. From a conceptual framework point of view, the difference among their fundamental control mechanisms is shown in Fig.7.

In the NL approach, the meanings of data values, the knowledge about database structures, and the relationship information among data components are included into the natural language processing module as the basic processing components. Moreover, several processing facilities in this module are dependent on the database contents completely because the database schema must be understood by such facilities in advance. Namely, the relationship between the application program(of the natural language processor) and DBMS is very tightly coupled though they are separated graphically in Fig.7(a). The natural language processor must be constructed one by one for different databases. While, in our semantic approach as illustrated in Fig.7(b), the DML handler interprets directly the natural-language-like queries through the semantic data model.

6. CONCLUSION

It is desirable that database management systems manage, by themselves, every information about the structures and contents of databases, and relationships among composite data elements. The semantic data model must have a representation and manipulation facility for such semantic information, in addition to the conventional structure information. The database management systems based on such semantic data models can provide more flexible and natural query forms to every end user.

Our semantic data model is not only based on the framework of the ER model, but also designed with the characteristics of the case grammar. Therefore, our model may be characterized as one of the semantic networks because of the exclusion of attributes and the introduction of roles. We have not yet evaluated the representation ability of our model sufficiently. Additionally, the features of the frame structure or the semantic network will be desirable to be introduced with respect to the concepts of generalization and specialization. However, our model is very adjustable to our objective: the interpretation of natural-language-like queries; and the representation and manipulation of semantic information about databases.

Now, we have developed a prototype database management system based on our semantic data model. The machine to be implemented is the workstation, which provides the LISP environment.

Acknowledgements --- We are grateful to Prof. Y.INAGAKI and Prof. J.TORIWAKI for their respective remarks, and wish to thank Mr.M.SUGIYAMA and Mr.N.HOCHIN, NTT Communications and Information Processing Laboratories of NTT Corporation, for their eager discussions.

This work was supported in part by the Telecommunications Advancement Foundation.

References

1) A.L.FURTADO & E.T.NEUHOLD: " Formal Techniques for Database Design", P.114, Springer-Verlag(1986).
2) B.J.GROOZ et.al.: " TEAM: An Experiment in the Design of Transportable Natural Language Interfaces ", Artificial Intelligence, 32, pp.173-243(1987).
3) P.CHEN: " The Entity-Relationship Model: Toward an Unified View of Data", ACM trans.on Database Systems, 1, 1 (1976).
4) J.PECKHAM & F.MARYANSKI: " Semantic Data Models ", ACM Computing Surveys, 20, 3, pp.153-189(1988).
5) D.JAGANNATHAN et.al: " SIM: A Database System Based on the Semantic Data Model", Proc.of SIGMOD'88, pp.46-55.
6) M.J.CAREY, D.J.DEWITT & S.L.VANDERBERG: " A Data Model and Query Language for EXODUS", Proc.of SIGMOD'88, pp.413-423.
7) C.FILLMORE: " The Case for Case ", on Universals in Linguistic Theory(ed. Bach & Harms), Holt, Rinchart & Winston(1968).
8) R.HULL & R.KING: " Semantic Database Modeling: Survey, Applications and Research Issues ", ACM Computing Surveys, 19, 3, pp.201-260(1986).