□ 研 究 論 文 □

韓 國 軍 事 運 營 分 析 學 會 誌
第 23 卷, 第 2 號, 1997. 12 . 31

Methodology for Extended Schema Representation in Database Integration

김 철 호*

Abstract

There have been several research efforts to support interoperability among multiple databases. In integrating multiple databases, we must resolve schema conflicts due to the heterogeneity in databases. To resolve these conflicts, not only meta-data for database schemas but also general knowledge expressing the real world meanings associated with the database schemas are required.

This paper presents a uniform representation method for relational schema and general knowledge base that is composed, among other things, of concept hierarchy and thematic roles in relationships, using the knowledge representation language L_k.

This representation method has a flexible descriptive power which facilitates concepts to be expressed at different levels of granularity and can describe semantically-rich expression in the knowledge base. Meta-data and related general knowledge expressed in L_k are used for input of the next step, such as conflict resolution and query processing of multiple databases.

*국방정보체계연구소

1. Introduction

Many databases have been constructed by different groups of users and organizations independently to meet their specific requirements during the last few decades. There are many applications requiring data not only from a single database but also from multiple databases in a local organization or networked world. Users of database systems want to find data and information without knowing which database has required data and where the database is located. For these reasons, interoperability or integration of multiple databases is one of the essential research issues in the database application area.

The two major approaches for database integration from separate databases are a unified schema(global schema)[3, 6] and a multidatabase language[10]. Following that, Wiederhold suggested the concept of a mediator, "a software module that exploits encoded knowledge about certain sets or subsets of data to create information for a higher layer of applications"[16].

In any case of integrating multiple databases, we must resolve conflicts such as: table-versus-table conflicts; attribute-versus-attribute conflicts; table versus attribute conflicts; and so on, due to the heterogeneity in databases[8]. To resolve these conflicts, not only meta-data for local database schemas but also general knowledge expressing the real world meanings associated with the database schemas are required.

In this paper, we use L_k[14, 15] as a meta-data and knowledge representation language. L_k is notable in offering a flexible descriptive power which facilitates concepts to be expressed at different levels of granularity and a versatile association mechanism which is capable of linking partially related concepts.

Extended schema representation methodology described in this paper has some advantages. Meta-data and related knowledge are represented in a uniform manner. Semantically-rich expressions are meshed with database schemas which have been already implemented. Semantic relations in the E R model disappear when they are applied to the tables in relational database, but this method can keep entities, attributes and relations of the E-R model continually without losing.

The rest of the paper is organized as follows: Section 2 gives related works; Section 3 gives database schema and related knowledge representation using L_k; Section 4 illustrates the database heterogeneity resolution process and query processing using this representation method. Finally, Section 5 concludes the paper and lists some future research works.

2. Related Works

A well-known family of database integration approaches are the global schema construction and the multidatabase language. Schema integration(global schema) procedures were defined and several multidatabase approaches were compared by Batini et. al.[2] and Sheth et. al.[13]. In the schema integration approach, either a global database administrator or a user has to first identify the semantic and other conflicts, and then construct a global schema to resolve these conflicts, before data in multiple databases could be accessed.

Problems of multidatabase approaches were summarized in Bright et. al.[3]. In the multidatabase language approach, the burden of integration is shifted from global database administrators to users and local database administrators. These methods trade a level of data independence (the global schema hides duplication, heterogeneity, and location information) for a more dynamic system and greater control over system information. While specifying the contexts associated with attribute values, meta -attributes are used in the mediation approach.

Following Wiederholds important concept of the mediator[15], Sciore et. al.[12] developed an array of methods to resolve automatically semantic conflicts in multidatabases. To specify the contexts associated with attribute values, meta -attributes are used. Lim and Shin[9] suggested a mediator-driven query handling approach in a multiple database environment using a knowledge base (concept hierarchies, general domain knowledge) in addition to meta-data of data sources.

At any rate, we should have a representation method for meta-data of data sources and related general knowledge to resolve conflicts in integrating databases. Sciore et. al.[12] used a semantic-value schema, a semantic-value specification and

C-SQL(Context-SQL) as an extended SQL to resolve data conflicts in multiple database. They did not describe the way of expressing the relationships among the attributes in different tables; such as foreign keys, the one-to-many correspondence relationship between relations in different databases, and the super-sub class relationships. Garcia-Solaco et. al.[7] adopted the BLOOM model to represent semantically rich structures of the schemas. BLOOM is a semantic extension of object-oriented models and its main constituents are objects and classes. The structure and behavior of their object members are described by class. The description of properties and relationships to other objects are entailed in the structure. This method considered the integration of classes, not attributes, and cannot resolve the structural conflicts. Bright et. al.[3] proposed the Summary Schemas Model(SSM) as an extension of multidatabase language systems. A summary schema represents a concise, more abstract description of the semantic contents of a group of input schemas. Rich semantic power of the SSM come from the linguistic knowledge represented in online dictionaries and thesauruses.

New approach in representing database schemas and related knowledge is similar to SSM. We use the frame based language L_k to represent local database schemas, and knowledge such as concept hierarchies and the case frame of verbs extracted from the online dictionary[5] of a natural language. The relational schema with the relationship of E-R model as well as related knowledge can be represented by L_k in the same manner. As far as we know, this approach is the only one that can represent database schema(table, attribute and relationship) and general domain knowledge in rich semantic expressiveness on a natural language level. All of database manipulation can be processed standard SQL without any extension.

3. DB Schema and Related Knowledge Representation

In this section, the language L_k is described briefly. Then the transformation rules to represent relational database schemas into L_k and a method to represent related domain knowledge using L_k are described.

3.1 The Language Lk

The goal of this subsection is to summarize the L_k which was developed for use in knowledge and data engineering areas, especially for the database application. L_k can express entities, states, and events in the real world meaning. The database schema and related knowledge are represented by using the basic unit of expression, called c-terms(concept-terms). A c-term is a frame-based structure of the following form:

$$l[c_1:t_1,\ c_2:t_2,\ \ldots,\ c_n:t_n],$$

where $n \geq 0$, l is called the head concept label, and for each i, $1 \leq i \leq n$, c_i is called a concept-connector, t_i, being itself a c-term, is called a target of the concept-connector, and $c_i:t_i$ is called a concept-restrictor. The head concept label can be a entity, state, or event label to denote the entity, state, or event in the real world. Only a specific set of connectors for each l is permitted to be included in its restriction.

L_k uses two special symbols, $\dot{:}$ and $\dot{\leq}$, to denote the instantiation and classification between two c-terms. For two c-terms, t_i and t_j, $t_i \ \dot{:}\ t_j$ means t_i is an instance concept of t_j, and $t_i \ \dot{\leq}\ t_j$ means t_i is a subclass concept of t_j.

In order to express complex real world concepts freely, three ways of extending the c-term class, such as indirect referencing, hyphenation, and fragmentation are introduced.

Eight c-term manipulation operations are defined in order to perform indirect associations and other reasoning steps systematically. They are upward label substitution, restrictor release, downward label substitution, restrictor introduction, restrictor inheritance, membership identification, concept-connector identification, and label-target rotation. These operations are used for generalization, abstraction, deduction, abduction, focus change and so on. Details of syntax, semantics, and c-term operations of the L_k and examples are described in [14, 15].

3.2 Relational Schema Representation in c-terms

This subsection suggests formal transformation rules to represent tables, attributes and foreign keys in a relational database.

Suppose that E is a table name in a database and it has a set of attributes, $\{a_1, a_2, ..., a_n\}$, $n \geq 1$. Some subset of attributes may describe a concept in the real world collectively. For example, {*street, city, state, zip-code, country*} describes "address" concept in the real world. We will call the set of those attributes the *composite attribute set.* An attribute may have a limited number of domain values which characterize the type of the table E. We will call those kinds of attributes the *type attributes.* For example, a *kind* attribute of the *item* which has "book", "microfilm", and "magazine" is a *type attribute.*

Transformation 1 (Composite Attribute Sets) If CAS is a composite attribute set, $\{a_1, ..., a_m\}$, in a table E, then CAS is transformed in the following form:

$$c\text{-}con{:}target[c\text{-}con_1{:}target_1, ..., c\text{-}con_m{:}target_m],$$

where *c-con* is a full meaningful word for the concept which is described by the CAS, *target* is the same as *c-con*. $c\text{-}con_i$, $1 \leq i \leq m$, is a full meaningful word for the concept which is described by the a_i and $target_i$ is a_i.

(Example 1) The attribute set {*str, ct, st, zip, country*} of the *employee* table will be transformed into *address:ADDRESS*[*street:STR, city:CT, zip-code:ZIP, country:COUNTRY*].

Transformation 2 (Type Attributes) If T is a type attribute in a table E and T has $\{v_1, ..., v_k\}$ as its domain values, then T is transformed in the following form:

$$c\text{-}con{:}target=\{c\text{-}con_1{:}target_1, ..., c\text{-}con_k{:}target_k\},$$

where *c-con* is the meaningful word for the concept denoted by T and target is T. *c-con*$_i$, $1 \leq i \leq k$, is the full meaningful word for the v_i and *target*$_i$ is v_i.
(Example 2) *kind* attribute of the *item* which has "book", "microfilm", and "magazine" as its value will be transformed into *kind:KIND={book:BOOK, microfilm:MICROFILM, magazine:MAGAZINE}*.

Transformation 3 (Attributes) If A is an attribute in a table E which is not belonging to a *composite attribute set* or is not a *type attribute*, then A is transformed in the following form:

c-con:target,

where *c-con* is a full meaningful word for the concept described by the attribute A and *target* is A.

(Example 3) *ssn* attribute of the *employee* table which denotes the social security number of a employee is transformed into *social security-number:SSN*.

Transformation 4 (Tables) If E is a table name in a database and it has a set of attributes $\{att_1, att_2, ..., att_n\}$, $n \geq 1$, then E is transformed in the following form:

l[**table-name**:E, *c-con*$_1$:*target*$_1$, ..., *c-con*$_m$:*target*$_m$], $1 \leq m \leq n$,

where l is a full meaningful word for the concept which the table name E denotes. *c-con*$_i$:*target*$_i$, $1 \leq i \leq m$, is constructed from an attribute or a subset of the attributes set through Transformation 1, 2, or 3.

(Example 4) Lets suppose that the relation table *item* has *attributes set*, {*i#*, *title*, *a name*, *str*, *city*, *state*, *country*, *subject*, *type*} and *type* attribute has the set of domain value, {book, cd-rom, microfilm}. The table is transformed into *LIBRARY-ITEM*[**table-name**:*ITEM*, *id:I#*, *title:TITLE*, *author-name:A NAME*,

subject:*SUBJECT*, *address*:*ADDRESS*[*street*:*STR*, *city*:*CITY*, *country*:*COUNTRY*], *type*:*TYPE*={*book*:*BOOK*, *cd-rom*:*CD-ROM*, *microfilm*:*MICROFILM*}].

The E-R model describes data as entities, relationships, and attributes. Not only entities and attributes but also relationships are applied to the tables in a relational database. A relationship set that is represented via a record containing the primary key of each of the entities and the attributes of the entity involved is used to represent the relationship. It is difficult to extract semantic relationships in reverse from the table in the relational database, because the property of relationships disappears after implementation.

However, c-terms can keep the relationships in E-R model via a state/event c-term using case frame and thematic roles in natural language[1]. The relationships represented in c-terms provide very useful knowledge when resolving the conflicts due to the foreign key relations and semantic heterogeneity in database integration. A knowledge base of relationships is constructed from an electronic dictionary and thesaurus. This is an excellent advantage in new approach.

Transformation 5 (Foreign Keys) If a set of attributes FK in relation R_1 is a *foreign key* and it refers to the relation R_2 which has a set of *primary keys* PK. Then the foreign key relation is described in c-terms. c terms whose head label is always ***FK*** is intended to denote the foreign key itself and the remaining c-term which is an event/state c-term is intended to denote the semantic meaning of the foreign key relation. The ***FK*** is constructed in the following form:

$$\boldsymbol{FK}[\boldsymbol{source}{:}R_1[fk_1, \ldots, fk_n], \boldsymbol{reference}{:}R_2[pk_1, \ldots, pk_n]],$$

where $n \geq 1$, $fk_i \in FK$ and $pk_i \in PK$.

The *event/state c-term* is constructed as following form:

$$l[t\text{-}role_1{:}l_1, \ldots, t\text{-}role_m{:}l_m],$$

where l is the state/event label to denote the semantic of the foreign key relation. $t\text{-}role_i$ is one of the thematic roles. l_i is corresponding to the *head label* of the c-terms into which the table R_i is transformed into c-terms through Transformation 4, respectively.

(Example 5) Figure 1 shows the E-R diagram that does not have relationship table after implementation. The foreign key relation is the attribute itself in the entity tables. The relational schema is from [6].

FK[***source***:*EMPLOYEE*[*dno*], ***reference***:DEPARTMENT[*dnumber*]] and *WORKS-FOR*[*agent*:*EMPLOYEE*, *beneficiary*:*DEPARTMENT*] are constructed for the foreign key *dno* of the table *EMPLOYEE* which refers to the *dnumber* attribute of the table *DEPARMENT*.

An employee works for a specific department.

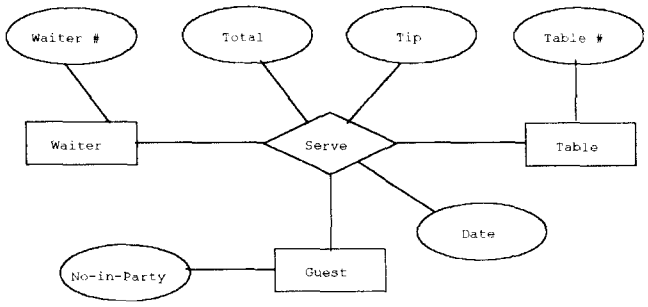
employee(fname, minit, lname, ssn, ..., dno)
department(dname, dnumber, mgrssn, mgrstartdate)

FK[***source***:*EMPLOYEE*[*dno*], ***reference***:DEPARTMENT[*dnumber*]]
WORKS-FOR[*agent*:*EMPLOYEE*, *beneficiary*:*DEPARTMENT*]

Figure 1. Foreign key relation(Case 1)

(Example 6) Figure 2 shows the E-R diagram from [4] that has relationship table after implementing in the relational database. The foreign key relations are attributes

in the relationship table.



A waiter serves a guest at a table.

WAITER(waiter#, name, sdate)
TABLE(table#, seats)
GUEST(guest#, noinparty)
SERVE(date, waiter#, guest#, table#, tip, total)

FK[***source***:*WAITER*[*waiter#*], ***reference***:TABLE[*table#*]]
FK[***source***:*WAITER*[*waiter#*], ***reference***:GUEST[*guest#*]]
FK[***source***:*GUEST*[*guest#*], ***reference***:TABLE[*table#*]]
SERVE[agent:WAITER, object:GUEST, location:TABLE]

Figure 2. Foreign key relations(Case 2)

Sample relational database modified from [8] and its transformation into the c-terms are depicted in Figure 3.

Table name	Attributes	General Description
item	(i#, title, a-name, subject, type, language)	Library items
lc-num	(i#, c-letter, f-digit, s-digit, cuttering)	Library of Congress number
publisher	(i#, name, tel, street, city, zip, state, country)	Publishers
lend-info	(i#, lend-period, library-use-only, checked-out)	Lending Information
checkout-info	(i#, id-num, hour, day, month, year)	Borrower and due date
employee	(fname, minit, lname, ssn, address, salary)	Faculties and staffs
student	(name, ssn, dob, address, sex, gpa, adv#, major)	Student

(a) relational database

LIBRARY-ITEM[table-name:ITEM, id:I#, title:TITLE, author-name:A-NAME,
subject:SUBJECT, type:TYPE={book:BOOK, cd-rom:CD-ROM, magazine:MAGAZINE},
language:LANGUAGE]
LIBRARY-CONGRESS-NUMBER[table-name:LC-NUM, id:I#, c-letter:C-LETTER,
f-digit:F-DIGIT, s-digit:S-DIGIT, cuttering:CUTTERING]
Has[agent:ITEM, object:LC-NUM]
FK[source:LC-NUM[id], reference:ITEM[id]]
PUBLISHER[table-name:PUBLISHER, id:I#, name:NAME, telephone-number:TEL,
address:ADDRESS[street:STREET, city:CITY, zip-code:ZIP, state:STATE,
country:COUNTRY]
Publish[agent:PUBLISHER, object:LIBRARY-ITEM]
FK[source:PUBLISHER[id], reference:LIBRARY-ITEM[id]]
LEND[table-name:LEND-INFO, id:I#, period:LEND-PERIOD,
library-use-only:LIBRARY-USE-ONLY, checked-out:CHECKED-OUT]
CHECKOUT[table-name:CHECKOUT-INFO, id:I#, lender-social-security-number:ID-NUM,
due-date:DUE-DATE[hour:HOUR, day:DAY, month:MONTH, year:YEAR]]
Borrow[agent:EMPLOYEE, object:LIBRARY-ITEM]
FK[source:CHECKOUT[id], reference:LIBRARY-ITEM[id]]
FK[source:CHECKOUT[lender-social-security-number],
reference:EMPLOYEE[social-security-number]]
FK[source:CHECKOUT[lender-social-security-number],
reference:STUDENT[social-security-number]]
EMPLOYEE[table-name:EMPLOYEE, name:NAME[first-name:FNAME,
middle-name-initial:MINIT, last-name:LNAME], social-security-number:SSN,
address:ADDRESS, salary:SALARY]
STUDENT[table-name:STUDENT, name:NAME, social-security-number:SSN, birth-date:DOB,
address:ADDRESS, sex:SEX=MALE[val:0], FEMALE[val:1], grade-point-average:GPA,
advisor-number:ADV#, major:MAJOR]

(b) c-term Representation

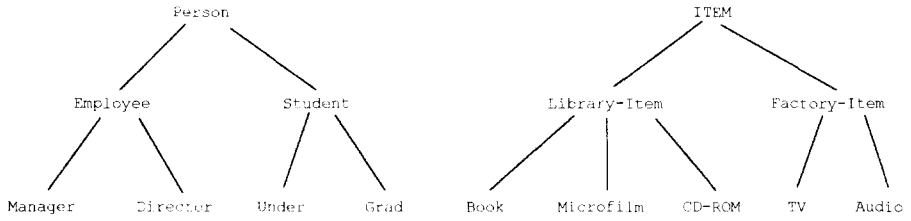
Figure 3. Library database and c-term representation

Figure 3. Library database and c-term representation

3.3 Domain Knowledge Representation in c-terms

Representing the general knowledge and database-specific domain knowledge are the inherent function of L_k. The collection of instantiation and classification assertion construct a concept hierarchy knowledge base. Entity, event, and state c-terms that we call the predication assertions described in 3.1 form the knowledge base also. Concept hierarchies and the related domain knowledge base are very useful resources in the conflict resolution process. Such kind of knowledge bases are constructed from a machine-readable electronic dictionary[5] and thesaurus prior to the conflict resolution. So, concept hierarchies and domain knowledge represented in c-terms have semantically-rich expressions on a natural language level.

Figure 4 depicts a *c-term* representation of a concept hierarchy knowledge base. General and database-specific domain knowledge is in Figure 5.



MANAGER ⊑ EMPLOYEE

DIRECTOR ⊑ EMPLOYEE

UNDER ⊑ STUDENT

GRAD ⊑ STUDENT

EMPLOYEE ⊑ PERSON

STUDENT ⊑ PERSON

BOOK ⊑ LIBRARY-ITEM

MICROFILM ⊑ LIBRARY ITEM

CD-ROM ⊑ LIBRARY-ITEM

TV ⊑ FACTORY-ITEM

AUDIO ⊑ FACTORY-ITEM

LIBRARY-ITEM ⊑ ITEM

FACTORY-ITEM ⊑ ITEM

/Entity/

BOOK[cover:HARD]

(Books whose cover is hard-bound.)

/State and Event/

OWN[agent:KIM, object:BOOK[title:"Intelligent Database",
author:PERSON[name:"Chung"]]]

(Kim owns the book "Intelligent Database" authored by "Chung".)

BORROW[agent:STUDENT[name:JOE, address:"Storrs"],
object:BOOK[title:"Visual Languages", author:"S. K. Chung"]]

(Student Joe who is living in Storrs borrowed the book "Visual Languages" authored by "S. K. Chung".)

Figure 5. General and domain knowledge representation in c-term

4. Application of the Extended Schema Representation

The extended schema representation method presented in the previous section can be applied to various database application areas such as a schema representation language, a multiple database integration, or a query manipulation. This section illustrates the global schema construction algorithm for schemas represented in c-terms, and illustrates a sample query processing. A formal and more complete description of the algorithm is beyond the scope of this paper, and can be found in [10].

4.1 Global Schema Construction

To integrate multiple databases, we devised a global schema construction algorithm. Some assumptions are imposed in the algorithm: no homonym is allowed;

a synonym dictionary is used; all of the table and attribute names are fully meaningful words. The condensed algorithm is described by the following steps:

1. Process the Composite Attribute Sets(CAS) to resolve the conflict in 1-to-n attributes, and attributes-versus-table conflicts.
2. Process the Type Attributes(TA) to build a local concept hierarchy.
3. Resolve the table-versus-table conflict.

3.1 Construct the list of the pairs of entities having the same head labels or the synonym head labels.

3.2 Construct a common entity c-term in the c-term library having only head labels, and mapping c-terms for each local schema having the same head label.

3.3 Process the common attributes(attribute-versus-attribute) conflict.

3.4 Process the attribute-versus-table conflict.

4. Put the tables having no corresponding tables into the c-term and mapping c-term library.
5. Build the domain concept hierarchies using the results from Step 1, 2 and the concept hierarchy in the knowledge base.
6. Accomplish event/state c-terms with local table names using the event/state c-terms stored in the knowledge base, and construct foreign key relationships between the tables.

Step 1 through 3 are similar to the approaches in works that have been done by other researchers. The remaining steps are a new attempt that use real world knowledge such as concept hierarchy and relationship in case frame of a natural language. The global schema constructed by the above algorithm is used for a database integration.

4.2 Query Processing

A users query to a single or multiple databases can easily be represented using a

A users query to a single or multiple databases can easily be represented using a simple GUI and c-terms. An example of the users query and its processing is shown below.

(Query) What is the address of the person who borrowed the book "Visual Language" authored by "S. K. Chung"?
For the above query to multiple databases, user input and system response will be as follows:

(1) Select an event/state c-term in the knowledge base.

```
BORROW[agent:PERSON, object:LIBRARY-ITEM]
```

(2) Click the PERSON.

The system now shows the concept hierarchy of the person, then select the PERSON and fill in

```
PERSON:
  address:?
  name:-
```

(3) Click the LIBRARY-ITEM.

The system now shows the concept hierarchy of the LIBRARY-ITEM, then select the BOOK and fill in

```
BOOK:
  author-name:"S. K. Chung"
  title:"Visual Language"
  subject:-
```

Query represented by c-terms can directly be translated into the SQL without any

5. Conclusion

There have been requirements for a standard form that is used for database administrators to put local database schema information in the database integration module. In this paper, we have presented an extended schema representation formalism which is developed to achieve a tight coupling of knowledge and data for multiple database environment. We also showed a global schema construction algorithm and a query processing in multiple database briefly as an example of c-terms application.

The approach presented in this paper has some advantages. Knowledge base such as the concept hierarchy and the relationship extracted from an electronic dictionary is a new approach. First, expressiveness of this schema and knowledge representation methodology has almost the same power of semantic richness found in a natural language level. Knowledge represented in c-terms performs an important role in conflict resolution in the database integration. Second, database schema and related domain knowledge are uniformly represented in c-terms. Third, this schema and knowledge representation methodology can be used for a single database environment as well. Finally, it also provides the easiness of query processing in the GUI environment.

We are doing research on finding conflicts which are not necessary to resolve when they are represented in c-terms. Further work need to be done for integrating the object-oriented database into a relational database environment. To do this, representing in c-terms will be extended for object-oriented database schemas. In order to cut down on intervention by a database administrator and user as much as possible in the database integration, an automatic algorithm for detection and resolution of conflicts will be studied. This future work will make for more intelligent processing in the database integration.

References

1. J. Allen, Natural Language Understanding, 2nd Ed., The Benjamin/Cummings Publishing Company, Redwood City, CA, USA, 1995.
2. C. Batini, M. Lenzerini and S. B. Navathe, A Comparetive Analysis of Methodologies for Database Schema Integration, ACM Computing Surveys, Vol. 18, No. 4, pp.323-364, December 1986.
3. M. W. Bright, A. R. Hurson and S. Pakzad, Automated Resolution of Semantic Heterogeneity in Multidatabases, ACM Transactions on Database Systems, Vol. 19, No. 2, pp.212-253, June 1994.
4. B. C. Desai, An Introduction to Database Systems, West Publishing Company, St. Paul, MN, USA, 1990.
5. EDR, EDR Electronic Dictionary Version 1.5 Technical Guide, TR2-007, Japan Electronic Dictionary Research Institute, Ltd. 1996.
6. R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 2nd Ed., The Benjamin/Cummings Publishing Company, Redwood City, CA, USA, 1994.
7. M. Garcia-Solaco, F. Saltor and M. Castellanos, A Structure Based Schema Integration Methodology, Proceedings of 11th International Conference on Data Engineering, Taipei, Taiwan, pp.505-512, March 1995.
8. W. Kim and J. Seo, Classifying Schematic and Data Heterogeneity in Multidatabase Systems, IEEE Computer, Vol. 24, No. 12, pp.12-18, December 1991.
9. J. S. Lim and D. G. Shin, Query Instance Generation for Multiple Database Systems, Proceedings of ISCA 6th International Conference on Intelligent Systems, Boston, USA, pp.135-139, June 1997.
10. J. S. Lim, C. H. Kim and D. G. Shin, Meta Data Construction Methodology for Multiple Databases, Submitted to IADT 98, Germany, July 1998.
11. W. Litwin, L. Mark and N. Roussopoulos, Interoperability of Multiple Autonomous Databases, ACM Computing Surveys, Vol. 22, No. 3, pp.267-293, September 1990.
12. E. Sciore, M. Siegel and A. Rosenthal, Using Semantic Values to Facilitate Interoperability among Heterogeneous Information Systems, ACM Transactions on

Database Systems, Vol. 19, No. 2, pp.254-290, June 1994.

13. A. P. Sheth and J. A. Larson, Federated Database Systems for Managing Distributed, Heterogeneous and Autonomous Databases, ACM Computing Surveys, Vol. 22, No. 3, pp.183-236, September 1990.
14. D. G. Shin, *Lk* : A Language for Capturing Real World Meanings of the Stored Data, Proceedings of IEEE 7th International Conference on Data Engineering, Kobe, Japan, pp. 738-745, April 1991.
15. D. G. Shin, An Expectation-Driven Response Understanding Paradigm, IEEE Transactions on Knowledge and Data Engineering, Vol. 6, No. 3, pp. 430-443, June 1994.
16. G. Wiederhold, Mediators in the Architecture of Future Information Systems, IEEE Computer, Vol. 25, No. 3, pp.38-49, March 1992.