A Flexible Paradigm for Semantic Integration in Cooperative Heterogeneous Databases

C. H. C. Leung, and S. Kaspi
Victoria University of Technology,
Ballarat Road, P.O. 14428, MMC, Melbourne 3001, Victoria, AUSTRALIA.
Email: clement@matilda.vut.edu.au Fax: +61 3 688 4050

ABSTRACT

An architecture for the effective semantic integration of pre-existing heterogeneous databases is presented. This architecture is based on self-descriptive data, and does not rely on any global conceptual schema nor controlled vocabulary. It makes use of a paradigm of man-machine cooperation and adopts a semi-automatic approach by involving the global user in decisions on data objects relevance. In this paradigm, there are three phases of query processing: meta-query processing, subquery evaluation, and data integration. It is based on the assumption that a fully automatic approach of devising elaborate algorithms to estimate user requirements is infeasible and error prone, since the mapping of given data items to corresponding ones in other databases may be unwieldily complex and involves a large number of decision points. Two possible sources of errors are identified in the processing of global queries: inclusion errors, where extraneous information are incorrectly incorporated into the final result; and exclusion errors, where relevant remote items are omitted in the answer. Within the present paradigm, a fully automatic system may also be obtained by adopting a subset of the architectural components for certain applications where the consequence of these errors is insignificant. Although this paper primarily focuses on relational database processing, the underlying method is sufficiently general for object-oriented post-relational databases as well as knowledge bases.

1. INTRODUCTION

A heterogeneous database is group of geographically distributed, independently developed and managed databases that overlap semantically which support user access to the combined data via a single request. However, the mechanisms to support the querying of heterogeneous databases is not well understood, and there is no accepted standard for developing these systems. Consequently the number of working heterogeneous databases is relatively small and are mostly experimental. Examples of heterogeneous databases include ADDS [4], CALIDA [15], DQS [2], MRDSM [18] Pegasus [25], and PRECI [10]. Due to the practical importance of distributed databases, there has been considerable interests in this area, and relevant studies addressing different aspects of cooperation are reported in [1, 3, 5, 6, 7, 9, 11-14, 16, 19-24, 26-33].

A major obstacle that prevents the effective cooperation between different databases is the inability to provide semantic integration of the underlying knowledge and data. Especially for pre-existing systems with rich semantics, automatic semantic integration will be neither feasible nor achievable. This work focuses on the use of self-descriptive data and extended data dictionaries, together with an interactive query model, to resolve semantic heterogeneity and enable effective integration. The operating environment consists of a number of autonomous heterogeneous databases possibly spanning organisational boundaries, with participating systems each providing an *export schema* and the associated database portions containing the relevant information. Cooperation as expressed in a single global query usually will result in a number of subqueries directed towards the relevant participating databases. Attention is primarily focused on highly structured data where, within a given database, precise meaning is attached to individual data items, and on queries where there is a

unique correct answer. It is not the aim of the present paper to consider evidential reasoning, and the provision of "best", but not necessarily correct answers. The organisation is based on the recognition that meta-query processing constitutes a substantial part of achieving effective cooperation, and such processing requires a combination of structured and unstructured data access. It is applicable to SQL and post-SQL (e.g. ODMG-93, SQL3) processing as well as generalisable to other knowledge-based languages, although for concreteness, we primarily focus on the relational model in this paper.

In [8], three paradigms for searching for data are identified: browsing, connection under logical implication and generalisation. Generalisation may be regarded as equivalent to what in [5] refer to as Global-Schema databases. Here, the responsibility for identifying synonyms, hypernyms/hyponyms, homonyms etc. rests with the global DBA's (Database Administrators). Given a domain, these DBA's devise a global schema to which they map each of the local schemas. This global schema would also contain a syntax and vocabulary for framing queries. This global schema is then replicated to each of the local nodes.

There are a number of disadvantages with this paradigm. Firstly, it constrains the form with which a query can be made as the syntax and vocabulary are defined by the global DBA's. From the implementation point of view, given that global schemas tend to be large, their replication over nodes is expensive. In addition, their maintenance is cumbersome since a change in a locally independent database normally requires changes in the global schema.

For these reasons, we eschew this paradigm in favour of one that falls more readily under the definition of connection under logical implication - that is, allowing the users to frame their own questions. A thesaurus is used to relate user expressed field names to those actually used in the databases. Where ambiguities exist, either because of the semantics of the data names or because of the framing of the questions, a dialogue between the user and the system is used to produce the final result. That is, our system is semi-automatic rather than fully automatic.

We consider that this man-machine cooperation is necessary for two reasons. Firstly, the combinatorial explosion that would result in allowing for all query possibilities is probably unmanageable under current technologies. Secondly, we consider that it is beyond the ambit of a computer system to make subjective interpretations of a user's intention. The salient features of our model are:

- It assumes that each of the independent databases in the system is relational.
- It assumes that all of the data in the system is self-descriptive.
- It does not use a global schema.
- It does not require a controlled vocabulary.
- Operation is semi-automatic rather than fully automatic.

2. QUERY MODEL

A query model includes not only the precise language of specifying a query (e.g. SQL), but also the user-interface and the sequence of interactions between the system and the enquirer leading to the final delivery of the answers to the latter. An important difference between query processing in the present situation and conventional query processing is the presence of three phases:

1. meta-query processing
2. multi-query evaluation
3. results consolidation.

These phases are represented diagrammatically in Figure 1. The first phase will typically include a series of requirements and information exchanges between the user and the system involving the use of structured and unstructured meta-data and supporting facilities. It is a process of data objects identification, clarification, and verification. In a conventional situation, only phase two is applicable which in general also consists of the formulation of query execution plans as well as query optimisation.

As a first step, the enquirer must ascertain the precise meaning of relevant data objects within a query, as these may have different meanings in different databases. This in general covers a range of possibilities and comprise a number of distinct subqueries directed to different databases. For an object-oriented database, named data objects within a query may belong to the following categories:

- class name,
- attribute name,

- path expression.

In the case of the relational model, the class name may be viewed as the table name, and the attribute name corresponds to columns within a table. The path expression can be regarded a type of pre-joining, and is related more to the processing aspects rather than the data structuring aspects. Before sending out each subquery, the user has to (a) identify which data objects among the collection of databases are relevant, and (b) determine whether the meaning of these objects correspond to the user's own definition. Data correlation exists at two levels:

(i) subquery results obtained from the same database,

(ii) the global integration of results based on the output of (i).

In an SQL environment, the first may be achieved by means of embedded SQL processing. In environments where disparate data models are involved, additional model translation between that of the enquirer and the target model before and after subquery processing is necessary. Prior to the issuing of subqueries, e.g. in SQL, a number of meta-queries to the export schema will have to be made, which may involve successive refinement and reformulation. The precise nature of interaction is governed by the structure and organisation of the export schema, and are detailed in the following sections. Having subsequently gathered the data from different databases, the individual answers have to be integrated and fused together to provide a single answer to the global query, and this will be part of the function of the query model. In Figure 1, the filtering of irrelevant data can be performed before or after the actual multi-query processing or both, and may be done in conjunction with data model translation if necessary.

3. EXPORT SCHEMA AND EXTENDED DATA DICTIONARY

A conventional data dictionary serves a wide variety of purposes, and are accessed by different personnel and software routines. Among its human users include:

- database administrators
- applications and systems programmers
- end users

In the present context, it is the database administrators who will decide on whether certain portions of the database and the associated data dictionary entries should be made public to allow participation in the distributed system. Unlike in a conventional system where access to the data dictionary by end users is typically infrequent, an extended form of the data dictionary will have to be made available online to the users for interrogation during query processing for ascertaining the meaning of the data. This form of the data dictionary may be regarded as passive since it is mostly used manually by people rather than automatically by the database management system.

In a conventional data dictionary, the information given may be categorised into:

- data information
- processing information
- installation/organisation information.

The processing information is primarily concerned with which application systems using which data items and files, and the relevant programs and transactions operating on particular data items. The installation/organisation information is concerned with the configuration set up, authorisation checks, security enforcement as well as backup and recovery procedures. Although these components are essential to the correct working of the database system, we shall mainly focus attention on the data content. The data content of the dictionary may be regarded as metadata, and as such they are essential for the proper processing of global queries. The data content also includes information on indices and access paths which are used by the query optimiser. As we are primarily interested in the logical and semantic aspects of data, we shall not be directly concerned with these physical aspects, although they will ultimately be required for the efficient processing of global queries.

In the relational model, a data dictionary will be implemented in the form of tables. Since both meta-data and applications data are represented in tabular form, we shall refer to the former as *system* tables, and the latter as *application* tables. There will be a local system table listing the set of all application tables in the local system, whether they are base relations or views. In addition, the set of all local data items will also be stored in another local system table. Assuming a particular application table has been identified, it is then necessary to identify the relevant data items. For

either tables or data items, the identification may proceed as follows. For concreteness, we shall concentrate on the identification of the latter. If a global query requests for an item with a specific name X, the aim is to identify all items in the distributed database $\Sigma = \{X1, X2, X3, ...\}$ which has the same *semantic content* as X. By a data item $X1$ having the same semantic content as the original item X, it is meant that $X1$ contains information referred to by X, either in its existing form or after suitable transformation. The user has to target two sets for investigation:

1. the set of items with names different from X but has the same semantic content ($\Sigma 1$),

2. the set of data items with the same name but have different semantic content ($\Sigma 2$).

The following gives the structure of the export schema of a local relational database. These may be extracted as views from the base tables in the data dictionary, and are enhanced with additional navigational information for supporting semantic integration.

Table_Set

Table_Name	*T_Pointer*	.....
T1	address1	.....
T2	address2	.....
.....	.....	.....
.....	.....	.....

Item_Set

Item_ Name	*Table_ Name*	*I_Pointer*	*T_Pointer*	.....
I1	T1	address3	address1	.....
I2	T1	address4	address1	.....
...	.....	.....	.....	.....
...	.....	.....	.....	.....

Here, the data items I1 and I2 are assumed to be columns of application table T1. The I_Pointer and T_Pointer fields are addresses pointing to the detailed description of the relevant item and table respectively, and these will be used in subsequent processing for the identification of the two sets $\Sigma 1$ and $\Sigma 2$. The structuring and processing of the description information is described in the next section. The T_Pointer is included in the Item_Set table to facilitate direct access to the table description without needing to go through the Table_Set and the accompanying join operation. Although the first two columns of Table_Set are included in the Item_Set table, there is other information held in the remaining columns of Table_Set not available in Item_Set. Moreover, while Table_Name is a primary key in Table_Set, it generally will have multiple value occurrence in the Item_Set table. The alias to name mapping is structured as follows, where T1', T1", and T1"' are aliases to application table T1.

Table_Aliases

Alias	*Name*
T1'	T1
T1"	T1
T1"'	T1
T2'	T2
...	...
...	...
...	...

In the data dictionary, an item will have an attribute name N, by which it is known within the local system. In addition to N, there are usually a set of aliases $S = \{N1, N2, N3, ...\}$ of the item which are alternative names for it, and are sometimes used within the local system. This mapping is stored as follows.

Item_Aliases

Alias	*Name*
N1	N
N2	N
N3	N
M1	M
M2	M
...	...
...	...

Given a specific data item X = 'bcdwxy' to be matched, the identification of the two sets $\Sigma 1$ and $\Sigma 2$ may proceed as follows. The Item_Set table will be searched for a match. Depending on the outcome of the match, the following may arise.

1. If a match is found, then the attribute description will be accessed through the I_Pointer, and the

table description will be accessed through the T_Pointer for subsequent meta-query processing. The set of relevant description pointers may be obtained as:

Set1
Select I_Pointer, T_Pointer
From Item_Set
Where Item_Name = 'bcdwxy'

2. If a match is not found, then the Item_Aliases table will be searched for a match. If a match is found, then the corresponding Name field will be read, which is used to probe the Item_Set table. In a similar way, the item description will be accessed through the I_Pointer, and the table description will be accessed through the T_Pointer. The set of relevant description pointers may be obtained as:

Set2
Select I_Pointer, T_Pointer
From Item_Set, Item_Aliases
Where Alias = 'bcdwxy'
And Name = Item_Name

Depending on the relative weights attached to exclusion error (i.e. the error of omitting items which ought to be included, see below), this step may also be performed even if a match is found in the preceding step.

3. If a match is not found in the Item_Aliases table, then the thesaurus will be invoked. Again, depending on the relative weights attached to exclusion error, this step may also be performed even if there are matches in the preceding two steps. All the relevant pointers obtained in these steps may be temporarily stored together for subsequent access to the description information.

The thesaurus will be incrementally compiled and updated, and a copy will be located at each site. Consistency of all thesauri across the network is not essential, although the ability to correctly identify the two sets $\Sigma 1$ and $\Sigma 2$ will partly depend on the competence of thesauri. A thesaurus will contain three classes of terms: broader, narrower, and related, which will be accessed by the query module for the formulation of subqueries. Initially, it may just contain the union of all the data item names and aliases of all participating databases. Other terms may be added with increased usage. The thesaurus is organised as three tables and is structured as follows.

RTerm

Term	*Related Term*
Car	Automobile
Car	Vehicle
A	A1
A	A2
A	A3
B	B1
...	...
...	...

NTerm

Term	*Narrower Term*
Part-time	Half-time
Part-time	Quarter-time
C	C1
C	C2
C	C3
D	D1
...	...
...	...

BTerm

Term	*Broader Term*
Programmer	Employee
Engineer	Employee
E1	E
E2	E
E3	E
F1	F
...	...
...	...

It is possible to structure tables NTerm and BTerm as a single table, and although the logical structures of these table are identical, their physical implementation may be quite different, and it is usually not possible to optimise performance for both directions of travel between broader and narrower terms simultaneously. In the NTerm table, the access is optimised for going from broad term to narrow term, whereas in the BTerm

table, the access is optimised for going from narrow term to broad term. Moreover, there are generally prefer directions of travel for specific terms, so that the opposite direction is seldom required. In such situations, the contents of NTerm and BTerm are not identical. The search argument X will be used to probe tables RTerm, NTerm, and BTerm respectively as follows.

SetR
Select Related_Term
From RTerm
Where Term = 'bcdwxy'

SetN
Select Narrower_Term
From NTerm
Where Term = 'bcdwxy'

SetB
Select Broader_Term
From BTerm
Where Term = 'bcdwxy'

After searching through the thesaurus, the set $\{SetR \cup SetN \cup SetB\}$ will be returned to the user for manual filtering, which will usually result in a smaller target set SetX for further narrowing down the investigation:

$$SetX = \{X1, ..., Xk\} \subset \{SetR \cup SetN \cup SetB\}.$$

The individual elements of $\{X1, ..., Xk\}$ will be used to probe the tables Item_Set, and Item_Aliases as before.

Suppose a set of items Σ' have been obtained. Associated with the two sets $\Sigma 1$ and $\Sigma 2$ are two possible sources of errors. The *exclusion error* E_x consists of excluding from the set Σ' items having the same semantic content as X. The *inclusion error* E_i consists of incorrectly including some extraneous items in the final result which have different semantic content as X (see Figure 2). In order to reduce the exclusion error, it is possible to carry out steps 2 and 3 in every situation, coupled also with partial matching in the specification of Set1 and Set2 queries using the **Like** operator, e.g.

Set1
Select I_Pointer, T_Pointer
From Item_Set
Where Item_Name **Like** '%bcd___'

Set2
Select I_Pointer, T_Pointer
From Item_Set, Item_Aliases
Where Alias **Like** '%wxy%'
And Name = Item_Name

Reducing the exclusion error may run the risk of raising the inclusion error and vice versa, and a good set of integration procedures will keep both errors low, and it is unlikely that this can be achieved with a fully automatic approach. Associated with the expansion in steps 2 and 3, there invariably will incur an associated filtering cost, and in situations where such cost outweighs the correctness of the query results, than a full expansion need not be carried out. Figure 3 shows the different aspects of the expansion process.

The set of all participating export schemas may be replicated at each site or stored in the same site as the local database in a distributed fashion. These organisations are shown respectively in Figure 4 and Figure 5. The first approach has the advantage of eliminating the need for involving the network in the course of meta-query processing and query clarification. Only properly formulated target queries need to be sent to the relevant databases, which are normally a subset of all participating databases. The second approach will result in more network traffic, but depending on the workload at individual sites, it is possible that a significant amount of concurrent meta-query processing may be carried out by the local databases, with a consequent reduction in processing for the originating site. With this approach, it will mean that *all* participating sites are involved in the meta-query processing of *every* global query, no matter where it originates, since it is usually not possible to limit considerations to a specific subset of the databases at the initial entry of a global query. In what follows, it is assumed that the first approach is used.

4. SELF-DESCRIPTIVE DATA

From the set of pointers obtained in the previous section (I_Pointer's and T_Pointer's), the relevant descriptions will be accessed by the user with a view of verifying whether the appropriate items or tables are indeed relevant to the present query. Again, we shall mostly focus on data items rather than tables. The description is a combination of structured and

unstructured information. The structured information consists of

1. a concise data definition of the item and its properties
2. relationship between the item in question and other data items in the database
3. the tables containing the data item.

Although the tables information may also be obtained directly by following the T_Pointer in the Item_Set table and also the Table_Set table, it is included here to provide an alternative path of access. Hence, it provides maximum access flexibility without requiring jumping back and forth between descriptions and the original tables.

Frequently, the structured information alone will not be sufficient to enable the user to determine whether the item has the same semantic content as the item X, and textual description with suitable cross-references will be required to establish this. Thus, in addition to the structured information, the description will be supplemented by a hypertext component. The hypertext will be used to support the unstructured interaction between the human enquirer and the system for the iterative clarification of the meaning of data objects, and a hypertext organisation will be able to provide the required degree of flexibility.

It is necessary to categorise a data object into (i) primitive, and (ii) derived. In the case of particular data items, the relationship between a primitive item and a derived item goes deeper than the distinction between base tables and views. The latter corresponds to data items which are derived from one or more other data items, and links will be available to move among them. For a primitive data item, informative textual description will be included detailing its meaning, and allowing invocation of definitions stored elsewhere in the dictionary. Precise transformation rules and formulae may be included in the structured part of the description in the form:

$$X = f_1\,(X1, X2, X3) = \ldots = f_k\,(X2, X8, X9)\ .$$

As can be seen, there may be multiple ways from which X may be derived so that the user may choose to query the representation most appropriate to his requirements.

In order to ascertain the meaning of an item, the users will often need to access *both* table information and item information, since the precise meaning of a data item may not be clear without knowing the table definition. A given requested item X may map to a single item X' in a different local database or may map to many items scattered over different tables in another local database, in which case all relevant tables must be accessed. For example, an item *No_units_produced* may be stored as a single yearly figure in one database while it is stored as a monthly figure divided into multiple regions recorded in multiple tables in another database. In extreme cases, item X may necessitate access to a matrix of values

$$\tilde{M} = \begin{pmatrix} X_{11} & X_{12} & \ldots & X_{1n} \\ X_{21} & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ X_{m1} & \ldots & \ldots & X_{mn} \end{pmatrix}$$

where these have to be functionally combined as $\varphi(\tilde{M})$ to produce an equivalent of X. For complex mapping and for items frequently accessed, it may be useful to establish a set of rules to facilitate the mapping and reduce the amount of exploratory search. These rules may be stored in the sites where the associated global queries originate. A possibility also exists where one or more of these tables may not be included in the export schema, in which case the original global query will need to be suitably adjusted by the user in order to obtain a meaningful answer. The use of man-machine cooperation is again essential in this situation where any automatic procedure is likely to lead to an unacceptable or unpredictable result.

Although he hypertext organisation is considered adequate for most applications, in some applications, e.g. CAD/CAM, where the use of graphics and images is essential to clarify the meaning of data, the use of hypermedia may be more appropriate. In particular, in certain types of measurements data, it may be necessary to provide actual pictures and diagrams illustrating the exact spatial points from which the measurements are taken. In other cases, it may be necessary to illustrate the detailed data collection process and data entry procedure so that the user may have sufficient information to determine the relevance of a data item.

Throughout the interaction, an *access list* is maintained, which records all data objects having the same semantic content as the original object together with their location details. This will be used by the subquery coordinator for the initial formulation of subqueries, and subsequently for the transformation, integration and presentation of the final results. The final integration of results will require the use of a consolidation operation which is particularly pervasive in distributed and parallel databases, and is discussed in [17] in the context of relational joins. Any redundant or duplicate information will have to be removed at this stage (see Figure 1).

5. Discussion and Concluding Remarks

The processing of global queries in cooperative heterogenous databases necessitates a considerable amount of intelligence in determining the relevance of data items scattered across different databases which were built with disparate applications objectives. In addition, the correct transformation and harmonious integration of subquery results also requires a substantial amount of processing. The intelligence in the present architecture can be regarded as being located at three points: (i) the human user, (ii) the thesaurus, and (iii) the subquery coordinator. By adopting an approach of man-machine cooperation through the use of self-descriptive data, it removes a substantial amount of complexity from the system, which at the same time ensures a high degree of accuracy in the final result. Another advantage of this organisation is that an individual database requires minimal effort to take part in the cooperation. No global conceptual schema needs to be built, and a participating database only needs to make available an export schema, the content of which should be mostly derivable from its existing data dictionary.

Although the present organisation is geared to a significant degree of human involvement, it is also able to support a fully automatic integration by simply following steps 1 to 3 detailed in Section 3 to incorporate *all* relevant items obtained without needing to retrieve the table or item descriptions. In doing so, the problem of meta-query processing is largely eliminated, as are the concomitant processes of navigation, clarification, and verification. Such a procedure will tend to give rise to a high inclusion error, and is only recommended for applications where the possibility or consequence of spurious inclusion is not significant. To limit the inclusion error, the expansion steps 2 or 3 may be omitted, but this will have an adverse effect on the exclusion error, and for a fully automatic system, the decision on which steps to include and under what conditions will have to be taken *a priori* since human consultation is excluded in the course of query processing. In the absence of human intervention, these errors are not easily measurable, and may creep in without their being detected in the result.

References

[1] Batini, C., Lenzerini, M., and Navathe, S.B. A Comparative analysis of Methodologies for Database Schema Integration, *ACM Computing Surveys*, Vol 18 no 4, Dec 1986, 322 -364.

[2] Belcastro, V., Dutowski, A., Kaminski, W, Kowalewski, M, Mallamici, C.L., Mezyk, S., Mostardi, T., Scorocco, F. P., StanizsKis, W, and Turco, G., An Overview of the Distributed Query System DQS, *Proceedings of the International Conference on Extending Database Technology-EDBT' 88*, Springer-Verlag, Berlin, 1988 170-189.

[3] Bell., D.A., Grimson, J.B. and Ling, D.H.O. EDDS - A System to Harmonize Access to Heterogeneous Databases on Distributed Nicros and Mainframes, *Info. Softw. Tech*, Vol 29, No 7, Sep 1987, 362-370.

[4] Breitbart, Y.J., and Tieman L.R. ADDS - Heterogeneous Distributed Database System, *Proceedings 3rd International Seminar on Distributed Data Sharing Systems*, North-Holland, Amsterdam, 1984, 7-24.

[5] Bright, M.W., Hurson, A.R. and Pakzad, S. Automated Resolution of Semantic Heterogeneity in Multidatabases, *ACM Trans. Database Systems*, Vol 19 No 2, June 1994, 212- 253.

[6] Bright, M. W., Hurson, A. Linguistic Support for Semantic Identification and Interpretation in multidatabases *Proceedings of the 1st International Workshop on Interoperability in Multidatabase Systems*, IEEE, Computer Society Press, Los Alamitos California, 1991, 306-313.

[7] Bright, M. W., Hurson, A. and Pakzad, S.H. A Taxonomy and Current Issues in Multidatabase

Systems, *IEEE Comput.*, Vol 25 No 3, March 1992, 50-60.

[8] D'Atri, A. and Tarantino, L. From Browsing to Querying, *Data Engineering*, Vol 12 No 2, June 1989, 46-53.

[9] Dayal, U. and Hwang, H. View Definition and Generalization for Database Integration in Multidatabase Systems, *IEEE Transactions on Software Engineering*, Vol 10, No 6, Nov 1984, 628-644.

[10] Deen, S.M. Amin, R.R. and Taylor, M.C. Implementation of a Prototype for PRECI, *Computer Journal*, Vol 30 No2, Feb 1987, 157-162.

[11] Fankhouser, P., Litwin, W., Neuhold, E.J. and Schrefl, M. Global View Definition and Multidatabase Languages - Two Approaches to Database Integration *Proceedings of the European Teleinformatics Conference EUTECO '88 on Research into Networks and Distributed Applications*, North-Holland, Amsterdam, 1988, 1069-1082.

[12] Fuhr, N. A Probablistic Framework for Vague Queries and Imprecise Information in Databases, *Proceedings of the 16th International Conference on Very Large Databases*, Morgan Kaufman, Los Altos, Cal., 1990, 696-707.

[13] Hurson, A.R. and Bright M.W. Global Information Access for Microcomputers, *Journal of Mini Micro Computer Applications*, Vol 10 No 2, 1991, 73-81.

[14] Hurson, A.R. and Bright M.W. Multidatabase Systems: An Advanced Concept in Handling Distributed Data, *Advances in Computers*, Vol. 32, Academic Press, San Diego, Cal, 1991, 217-268.

[15] Jakobsen, G., Piatesky-Shapiro, G., Lafond, C., Rajinkanth, M. and Hernandez, J. CALIDA: A System for Integrated Retrieval from Multiple Heterogeneous Databases, *Proceedings of the 3rd International Conference on Data and Knowledge Bases*, Morgan Kaufman, San Mateo, Cal, 1988, 3-18.

[16] Leung, C. H. C. and K. Wolfenden, Analysis and optimisation of data currency and consistency in replicated distributed databases, *Computer Journal*, Vol. 28, No. 5, 518-523, 1985.

[17] Leung, C. H. C. and H. T. Ghogomu A high-performance parallel database architecture, *Proc. 7th ACM International Conference on Supercomputing*, Tokyo, July 1993, 377-386.

[18] Litwin, W An Overview of the Multidatabase System MRDSM, *Proceedings of ACM Annual Conference*, ACM Press, New York, 1985.

[19] Litwin, W. and Abdellatif, A. Multidatabase Interoperability, *IEEE Comput.*, Vol. 19 no12, Dec 1986, 10-18.

[20] Litwin, W. and Abdellatif, A. An Overview of the Multidatabase Manipulation Language MDSL, *Proceedings IEEE*, Vol 75 no 5, May 1987, 621-632.

[21] Litwin, W. and Zeroual, A. Advances in Multidatabase Systems Integration, *Proceedings of the European Teleinformatics Conference EUTECO '88 on Research into Networks and Distributed Applications*, North-Holland, Amsterdam, 1988,1137-1151.

[22] Landers, T. and Rosenberg R., An Overview of Multibase Distributed Data Bases, *Proceedings of 2nd International Symposium on Distributed Databases*, H.J. Schneider Ed., North-Holland, Amsterdam, 1982, 153-183.

[23] Motro, A. Accomodating Imprecision in Database Systems: Issues and Solutions, *SIGMOD Record*, Vol 19 no4, Dec 1990, 69-74.

[24] Motro, A. A Trio of Database User Interfaces for Handling Vague Retrieval requests, *Data Engineering*, Vol 12 no 2, June 1989, 54-63.

[25] Rafii, A., Ahmed, R., Desmedt, P., Kent, B., Ketabchi, M., Litwin, W., and Shan, M. Multidatabase Management in Pegasus, *Proceedings of the 1st International Workshop on Interoperability in Multidatabase Systems*, IEEE Computer Society Press, Los Alamitos, Cal, 1991, 166-173.

[26] Reddy, M. P., Siegel, M., and Gupta, A. Towards an Active Schema Integration Architecture for Heterogeneous Database Systems, *International Workshop on Research Issues On Data Engineering: Interoperability in Multidatabse Systems* (RIDE-

IMS93, Vienna, Austria), IEEE, New York, 1992, 178-183.

[27] Rosenthal, A. and Siegel, M. An Architecture for Practical Metadata Integration. *Workshop on Information Technologies Systems*, Cambride, Mass, Dec 1991, 98 - 106.

[28] Sciore, E., Siegel, M. and Rosenthal A. Using Semantic Values to Facilitate Interoperability Among Heterogenous Information Systems, *ACM Trans. Database Systems*, Vol 19 No 2, June 1994, 254-290.

[29] Sheth, A.P. and Larson J.A. Federated Database Systems for Managing Distributed Heterogenous and Autonomous Databases, *ACM Computing Surveys*, Vol 22 No 3, Sep 1990, 183-236.

[30] Siegel, M., and Madnick, S., Schema Integration Using Metadata, *NSF Workshop on Heterogeneous Databases*, Evanston, Illinois, 1989.

[31] Siegel, M., Madnick, S. and Gupta , A. Composite Information Systems: Resolving Semantic Heterogeneities, *Workshop on Information Technology Systems*, Cambridge, Mass, Dec 1991, 125-140.

[32] Templeton, M., Brill, D., Dao, S.K., Lund, E., Ward, P., Chen, A.L.P. and Mac Gregor, R. Mermaid - A Front-End to Distributed Heterogeneous Databases, *Proceedings IEEE*, Vol 75 no 5, May 1987, 695- 708.

[33] Thomas, G., Thomson, G.R., Chung, C.W., Barkmeyer, E., Carter, F., Templeton, M., Fox, S. and Hartman, B. Heterogeneous Distributed Database Systems for Production Use, *ACM Computing Surveys*, Vol 22 No 3, Sep 1990, 237-266.

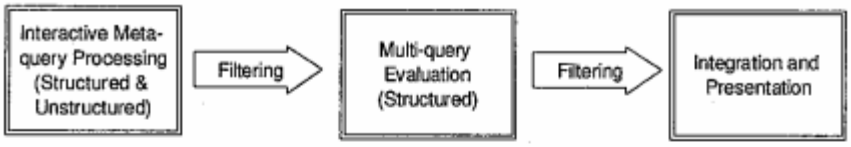

Figure 1

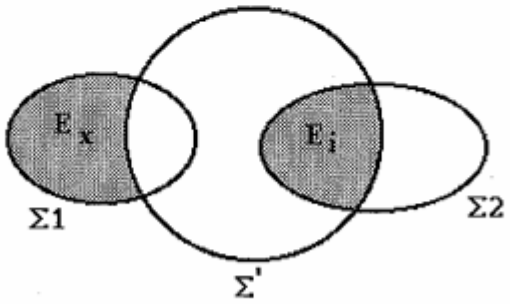

Figure 2

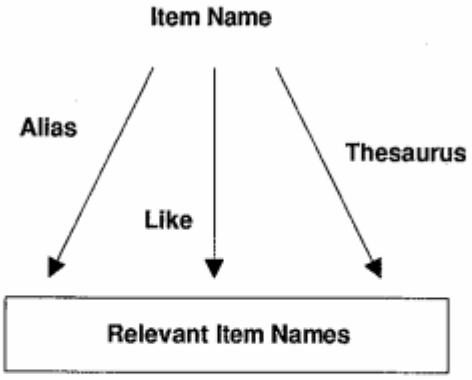

Figure 3

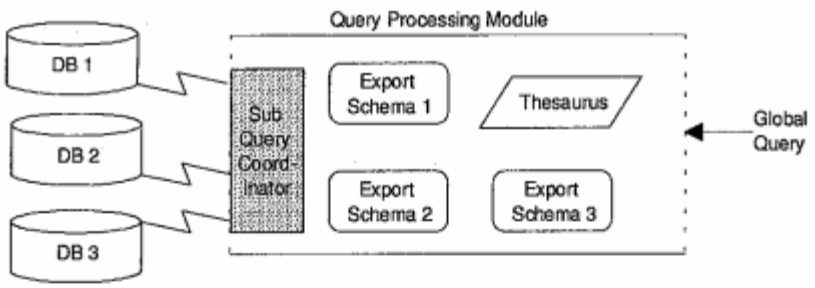

Figure 4

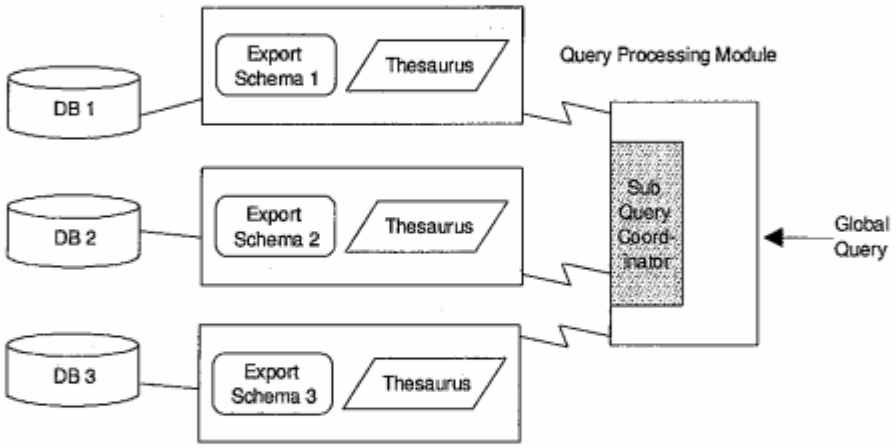

Figure 5