Data Integration in Digital Libraries: Approaches and Challenges

Ismail Khalil Ibrahim, Wieland Schwinger
Software Competence Center Hagenberg, Hauptstrasse 99, A-4232 Hagenberg, Austria
{ismail.khalil-ibrahim, wieland.schwinger}@scch.at

Abstract

Data integration is a central issue in digital libraries to facilitate the access and manipulation of the highly distributed, heterogeneous, and dynamic collection of information sources. In this paper, we survey the data integration approaches that meet the digital library requirements and identify the specific issues that contribute to the complexity of data integration in digital libraries.

1. Introduction

Information is power and digital libraries are built to provide a unified infrastructure for supporting the creation of information sources, facilitating the movement of information across global networks and allowing effective and efficient interaction among knowledge producers, librarians, and information seekers [2].

A digital library is a vast collection of entities stored and maintained by multiple information sources including databases, image banks, file systems, email systems, the Web, and applications providing structured or semi-structured data.

The dramatic growth of digital libraries in recent years has not only simplified the access to existing information sources but also initiated the creation of numerous new sources. Paradoxically, this growth has made the task of finding, extracting and aggregating relevant information not easier.

This is because most of the information systems underlying digital libraries are physically distributed, heterogeneous in the way how information is stored, organized and managed, and comprise heterogeneous software and hardware platforms on which they reside. Additionally, they are autonomous in the sense that the content and format of data are determined by the organization owning the data not by the user [13].

Furthermore, the data in these information sources is mainly of composite multimedia nature comprising different media types such as text, video, images, and audio, and dynamic in the sense that the sources update both the content and the form of the data on their own, possibly on unknown schedule.

The goal of data integration system is to provide users with a uniform interface to access, relate, and combine data stored in multiple, autonomous, and possibly heterogeneous information sources.

The most important advantage of a data integration system is that it enables users to focus on specifying what they want, rather than thinking about how to obtain the answers. Consequently, this frees users from the tedious tasks of finding the relevant information sources, interacting with each source in isolation using a particular interface and combining data from multiple sources.

Data integration requires that the differences in modeling, semantics and capabilities of the sources together with the possible inconsistencies be resolved. More specifically, the major issues that make integrating such data difficult include [13]:

Heterogeneity of information sources: Each source that is to be integrated might model the world in its own way. The representation of data of the similar semantics might be quite different in each data source. For example, each might be using different naming conventions to refer to the same real world object. Moreover, they may contain conflicting data.

Autonomy of information sources: usually information sources are created in advance of the integrated system. In fact, most of the time they never know that they are part of integration. They can take decisions independently and they cannot be forced to act in certain ways. As a natural consequence of this, they can also change their data or functionality without any announcement to the outside world.

Query correctness and performance: queries to an integrated system are usually formulated according to the unified model of the system. These queries need to be translated into forms that can be understood and processed by the individual data sources. The mapping should not cause incorrectness in query results. Also, query performance needs to be controlled, as there are many factors that can degrade it. These include the existence of a network environment, which can cause communication delays, and the possible unavailability of the data sources for answering queries.

Database community has much to offer to enable users to seamlessly and transparently access digital libraries.

Gateways, protocols like ODBC and standards like XML are gaining popularity in digital libraries because they potentially solve the issues of distribution and heterogeneity.

In this paper, we survey the data integration approaches that meet the digital libraries requirements and identify the specific issues that contribute to the complexity of data integration in digital libraries.

2. Motivational Scenario

Locating information sources by using web search engines or subject directories is just the first step in a long process. Users must go to these information sources, searching or browsing each one in isolation before moving on to the next and so on. This laborious task of resources discovery and retrieval is absolutely anachronistic.

As an example let us consider the task of searching bibliographic records of technical reports from digital libraries available on the WWW.

There are different online collections available on the www, which provide such information. Unfortunately, up to now we don't have a unified catalog of bibliographic records like the OCLC [2] for digital libraries. Probably the best example of such a digital library containing technical reports is the *Networked Computer Science Technical Reports Library* (http://cs-tr.cs.cornell.edu/) (NCSTRL)1 but this also provides only information of the sites that have installed the same software package (Dienst) and created their bibliographic records using the same format (RFC 1807).

Let us suppose that there are two information sources providing listings of technical reports that can be downloaded from the WWW. The first provides information about titles, authors, and subjects of the technical reports. The second provides information about the years, titles, and the URL from which you can download the report. Suppose we want to find which technical reports authored by Stéphane Bressan can be downloaded from MIT web site as well as their subjects.

None of these two digital libraries in isolation can answer this query. However, by combining data from multiple sources, we can answer queries like this one, and even more complex ones. To answer our query, we would first send a query q_1 to the first site (S_1) to retrieve the titles and subjects of all the technical reports authored by Stephane Bressan, and then send another query q_2 to the second site (S_2) to retrieve the titles of all the technical reports that can be downloaded from MIT site. Finally, join the results of the queries q_1 and q_2 to get the answer to our query.

The reader may start thinking about the network performance issues, the high connection overheads, high computation time, financial charges, and temporary unavailability, as well as other problems of manipulating sources with different data models, semantics, and access methods [1].

3. Approaches to Data Integration in Digital Libraries

A lot of research has been conducted in the field of data integration systems, and researchers have approached the problem from multiple points of view to provide a global schema that facilitates transparent access to distributed data, to integrate views, to facilitate interoperability without creating new objects, and to develop a new application with new objects that encompass existing applications [16].

The integration of data sources in digital libraries poses many challenges due to the differences in the data management systems (e.g., different vendors), in the data models (e.g., relational, network, ER, object-oriented etc.), in the query and data manipulation languages, in the data types (e.g., text, graphics, multimedia, hypermedia, etc.), in the format (e.g., structured, semistructured), and in the semantics.

Database interoperability [15, 18, 3, 17] is the ability of distributed, heterogeneous databases, which are independently created and administrated and have different semantics and schemas to cooperate and interoperate in a transparent way to the user while maintaining their autonomy and objectives.

The requirements and objectives for database interoperability are stated to be distribution transparency, heterogeneity transparency, no change to the existing database systems and applications, easy evolution of the system, execution of retrieval and updates, and performance comparable to homogenous distributed systems.

Different approaches and techniques [8, 23] have been proposed by the research community for data integration and several systems have been built with the goal of answering queries using a multitude of data sources.

Two common approaches have been advocated to building data integration systems [20]. The first approach is refereed to as virtual approach (Figure 1) to data integration. In the virtual approach, the user or the application poses the query. The data integration system accept the query, determine which set of information sources is capable to answer the query and generate the appropriate query plans for each information source. On obtaining the results from the information sources, the data integration system performs the appropriate translation,

[1] NTSTRL provides one-stop shopping for computer science technical reports from hundreds of institutions around the world by requiring that each site install the same software package (Dienst) and create bibliographic records using the same format (RFC 1807). At any of the NCSTRL sites, the search is sent simultaneously to all sites, then those sites search their local indexes and return their results, the results are received and collected by the initiating site, and they are displayed to the user. When a particular record or report is requested, the remote server that has the report responds to the request.

filtering and merging of the information and return the final answer to the user or application.

This process also may be referred to as a mediated approach, since the part that decomposes queries and combines results is often called the mediator.

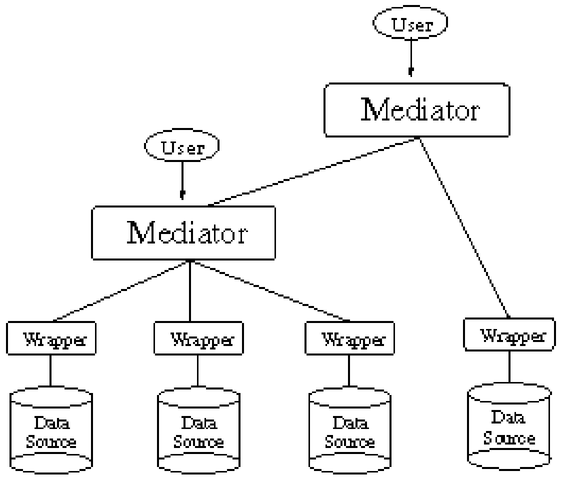

Figure 1. The virtual approach architecture

The second approach is called materialization approach (Figure 2) to data integration. In this approach information from each source that may be of interest to specific users or applications is extracted in advance, translated and filtered as appropriate, merged with relevant information from other sources and stored in a (logically) centralized repository.

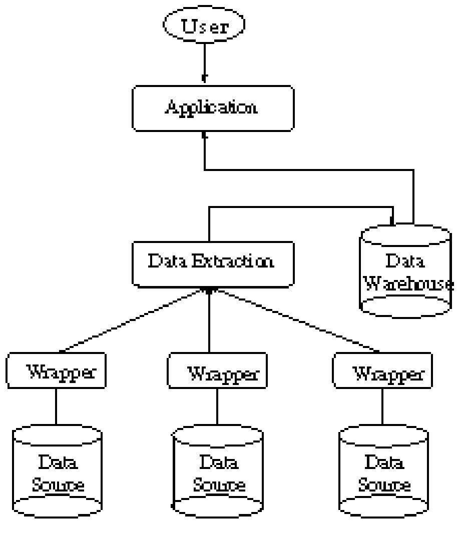

Figure 2. Materialization approach architecture

When a user or an application poses a query, the query is evaluated directly at the repository without accessing the original information sources. This approach is also referred to as data warehousing since the repository serves as a warehouse storing the data of interest.

The virtual approach to integration is appropriate when (a) the number of information sources is very large, (b) the data is changing very rapidly, (c) for clients with unpredictable needs and (d) for queries that operate over vast amounts of data from very large numbers of information sources (e.g., the World Wide Web). However, the virtual approach may incur inefficiency and delay in query processing, especially when (a) queries are issued multiple times, (b) information sources are slow, expensive or periodically unavailable and (c) significant processing is required for the translation, filtering and merging steps. In cases where information sources do not permit ad hoc queries, the virtual approach is simply not feasible. In the warehousing approach, the integrated information is available for immediate querying and analysis by clients. Thus, the warehousing approach is appropriate for (a) clients requiring specific predictable portions of the available information, (b) clients requiring high query performance but not necessarily over the most recent state of the information; (c) environments in which native applications at the information sources require high performance (large multi-source queries are executed at the warehouse instead) (d) clients wanting access to private copies of the information so that it can be modified, annotated, summarized and so on and (e) clients wanting to save information that is not maintained at the sources However, the data warehousing approach may incur that the warehouse should be updated each time the data is changed. For these reasons, most of the recent research has focused on the virtual approach in building data integration systems and especially in building web data integration systems.

Figure (3) illustrates the different components of this system [21].

Users of data integration systems do not pose queries directly in the schema in which the data is stored. Instead the user poses queries on a *mediated schema* (often referred to as a *global schema*), which describes the contents of data sources and exposes the aspects of the data that might be of interest to the user [14].

A mediated schema is a set of virtual relations (in the sense that they are not actually stored anywhere), which are designed for a particular data integration application. As a consequence, the data integration system must first reformulate the user query into a query that refers directly to the schemas in the sources. In order for the system to be able to do this, it needs to have a set of source descriptions, specifying the semantic mapping between the relations in the sources and the relations in the mediated schema. These descriptions specify the relationship between the relations in the mediated schema and those in the local schemas of the sources. The description of a data source specifies its contents (contains technical reports in our motivational example), attributes (titles, subjects), constraints on its contents (access methods), completeness and reliability, and finally its query processing capabilities (can perform selections or can answer arbitrary SQL queries).

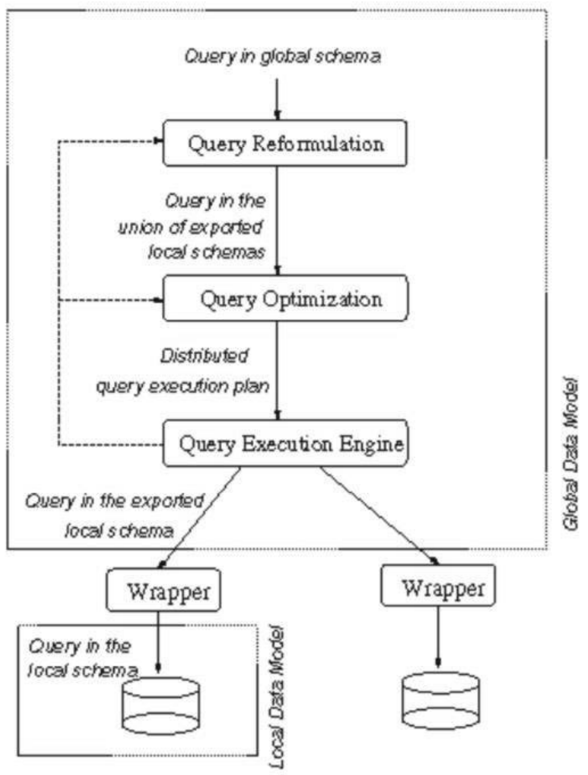
Query in global schema
Query Reformulation
Query in the union of exported local schemas
Query Optimization
Distributed query execution plan
Query Execution Engine
Query in the exported local schema
Global Data Model
Wrapper
Wrapper
Query in the local schema
Local Data Model

Figure 3. Components of data integration system

After the minimal set of data sources has been selected for a given query, a key problem is to find the optimal query execution plan for this query. The query execution plan is an imperative program that specifies exactly how to evaluate the query. In particular, the plan specifies the order in which to perform the different operations in the query (join, selection, projection), a specific algorithm to use for each operation (for example sort–merge join, hash–join) and the scheduling of different operators. Typically, the optimizer selects a query execution plan by searching a space of possible plans and comparing their estimated cost. To evaluate the cost of a query execution plan the optimizer relies on extensive statistics about the underlying data, such as sizes of relations, sizes of domains and selectivity of predicates. Finally, the query execution plan is passed to the query execution engine, which evaluates the query.

The system communicates with the remote data sources through wrappers. A wrapper is a program that is specific to a data source, whose task is to translate data from the format of the source to a format that can be manipulated by the data integration system. For example, if the data source is a Web site, the task of the wrapper is to translate the query to the source′s interface and when the answer is returned as an HTML document, it needs to extract a set of tuples from that document.

4. Complexity of Data Integration Systems in Digital Libraries

In addition to the problems encountered in building heterogeneous database integration systems, data integration in digital libraries have to deal with the huge amount of multimedia objects, the no or little metadata about the characteristics of these objects and the larger degree of autonomy of sources with none of the processing capabilities typically provided by databases, such as the ability to answer queries, or perform joins [9].

We classify the problems addressed in the area of data integration systems as follows [9, 14]:

4.1 Data modeling and query transformation

As described in the previous section, one of the main differences between a data integration system and a traditional database system is that users pose queries in terms of a mediated schema. The data, however, is stored in the data sources, organized under local schemas. Hence, in order for the data integration system to answer queries, there must be some description of the relationship
between the source relations and the mediated schema. The query processor of the integration system must be able to reformulate a query posed on the mediated schema into a query against the source schemas.
Broadly speaking, two main approaches have been considered for describing data sources:

A. *Global As View* (GAV)

In the GAV approach [14], for each relation R in the mediated schema, we write a query over the source relations specifying how to obtain R's tuples from the sources.

Back to our motivational example, suppose we have two sources DB_1 and DB_2 containing titles, authors and years of technical reports. We can describe the relationship between the sources and the mediated schema relation *ReportYear* as follows:

$$DB_1\,(id, title, author, year) \rightarrow ReportYear(title, year).$$
$$DB_2\,(id, title, author, year) \rightarrow ReportYear(title, year).$$

If we have a third source that shares technical reports identifiers with DB_1 and provides reports subjects, the following sentence describes how to obtain tuples for the *ReportSubject* relation:

$$DB_1\,(id, title, author, year)\ \&\ DB_3\,(id, subject) \rightarrow ReportSubject(title, author, subject)$$

Query reformulation in GAV is relatively straightforward. Since the relations in the mediated schema are defined in terms of the source relations, we need only unfold the definitions of the mediated schema relations. For example, suppose our query is to find subjects of 1997 reports:

q (*title, subject*) ← *ReportYear* (*title,* 1997), *ReportSubject*(*title, subject*)

Unfolding the descripitions of ReportYear and ReportSubject will yield the following queries over the source relations: (the second of which will obviously be deemed redundant)

q(*title*; *review*)← DB_1 (*id, title, author, year*), DB_3 (*id, subject*)
q(*title*; *review*) ← DB_1 (*id, title, author, year*), DB_2 (*title, author, year*), DB_3 (*id, subject*)

B. *Local As View* (LAV)

The LAV approach is the opposite of GAV. Instead of writing rules whose consequents are relations in the mediated schema, the rules contain a conjunction of atoms over the mediated schema in the consequent, and an atom of the source relation in the antecedent. That is, for every data source *S*, we write a rule over the relations in the mediated schema that describes which tuples are found in *S*.

Suppose we have two sources: (1) V_1, containing titles, years and authors of technical reports after 1990, and (2) V_2 containing technical reports subjects.

In LAV, we would describe these sources by the following sentences (variables that appear only on the right hand sides are assumed to be existentially quantified):

S_1: V_1 (*title, year, author*)→ *Report*(*title, year, author, institution*) & *NUS*(*author*) & *year* ≥ 1990 & *subject* = "*Database*".

Query reformulation in LAV is trickier than in GAV, because it is not possible to simply unfold the definitions of the relations in the mediated schema. In fact, the reformulation problem here leads to a new inference problem, which can be explained intuitively as follows. Because of the form of the LAV descriptions, each of the sources can be viewed as containing an answer to a query over the mediated schema (the one expressed by the right hand side of the source description).

Hence, sources represent materialized answers to queries over the virtual mediated schema. A user query is also posed over the mediated schema. The problem is therefore to find a way of answering the user query using only the answers to the queries describing the sources.

For example, suppose our query asks for subjects of technical reports on database written after 1990:

q(*title*; *review*) ← *Report* (*title, year, author, Database*), *year* ≥ 1990, *Subject*(*title, subject*).

The reformulated query on the sources would be

q'(*title, subject*) ← V_1(*title, year, author*), V_2(*title, subject*).

The LAV reformulation problem has received significant attention because of its relevance to other database problems, such as query optimization [7], maintaining physical data independence [22, 19], and data warehouse design.

The GAV and the LAV strategies can be qualitatively or quantitatively [6] compared in terms of their adequacy (a) to model a particular integration situation, (b) to cope with autonomy of the sources (sources changing their exported schemas, joining or leaving the network) and their ability (c) to answer queries.

The main arguments against the GAV strategy are that (a) it may not be able to model integration situation where sources are missing to build a complete world view; (b) it may stop to offer a complete global view as some sources become unavailable or services are disrupted [5]. In favor of GAV, it can be argued that most practical applications will require sufficiently simple global schema (unions) to avoid such difficulties and that there might be enough economic incentives in participating to network to convince the sources' managers to play the game. The strength of GAV is that, if the modeling is successful, (c) all queries on global schema are guaranteed to be answered and a complete answer can be constructed.

The LAV strategy, conversely, is designed to cope with a dynamic, possibly incomplete set of sources. The counterpart of this flexibility is that all queries may not be answered or only an incomplete answer can be found. It can be argued in favor of LAV that in large information infrastructures such as WWW, complete answers are rarely expected or needed by the users: "*better some answers than no answers*".

4.2 Completeness of data sources

In general, sources that we find in digital libraries are not necessarily complete for the domain they are covering. For example, a bibliography source is unlikely to be complete for the field of Computer Science. However, in some cases, we can assert completeness statements about sources. For example, the DB&LP Database (http://www.informatik.uni-trier.de/ley/db/) has the complete set of papers published in most major database conferences. Knowledge of completeness of a digital library can help a data integration system in several ways. Most importantly, since a negative answer from a complete source is meaningful, the data integration system can prune access to other sources.

4.3 Query processing capabilities

From the perspective of the web data integration system, the digital libraries appear to have vastly differing query processing capabilities [4].

The main reasons for the different appearance are [9] (1) the underlying data may actually be stored in a structured file or legacy system and in this case the interface to this data is naturally limited, and (2) even if the data is stored in a traditional database system, the digital library may provide only limited access capabilities for reasons of security or performance. To build an effective data integration system, these capabilities need to be explicitly described to the system, adhered to, and exploited as much as possible to improve performance. We distinguish two types of capabilities: negative capabilities that limit the access patterns to the data, and positive capabilities, where a source is able to perform additional algebraic operations in addition to simple data fetches. The main form of negative capabilities is limitations on the binding patterns that can be used in queries sent to the source. For example, it is not possible to send a query to the NCSRL asking for all the technical reports in the database and their subjects. Instead, it is only possible to ask for the subject of given technical report, or to ask for the set of technical reports authored by a particular author.

Positive capabilities pose another challenge to a data integration system. If a data source has the ability to perform operations such as selections and joins, we would like to push as much as possible of the processing to the source, thereby

Hopefully reducing the amount of local processing and the amount of data transmitted over the network.

4.4 Query optimization

Many works on web data integration systems have focused on the problem of selecting a minimal set of web sources to access, and on determining the minimal query that needs to be sent to each one. However, the issue of choosing an optimal query execution plan to access the web sources has received relatively little attention in the data integration literature [11], and remains an active area of research. The additional challenges that are faced in query optimization over sources on the WWW is that we have few statistics on the data in the sources, and hence little information to evaluate the cost of query execution plans.

4.5 Query execution

Even less attention has been paid to the problem of building query execution engines targeted for web data integration. The challenges in building such engines are caused by the autonomy of the data sources and the unpredictability of the performance of the network.

In particular, when accessing web sources we may experience initial delays before data is transmitted, and even when it is, the arrival of the data may be bursty. The work described in [5] has considered the problem of adapting a query execution plans to initial delays in the arrival of the data.

4.6 Wrappers

Recall that the role of a wrapper is to extract the data out of a web site into a form that can be manipulated by the data integration system. For example, the task of a wrapper could be to pose a query to a web source using a form interface, and to extract a set of answer tuples out of the resulting HTML page. The difficulty in building wrappers is that the HTML page is usually designed for human viewing, rather than for extracting data by programs. Hence, the data is often embedded in natural language text or hidden within graphical presentation primitives. Moreover, the form of the HTML pages changes frequently, making it hard to maintain the wrappers. Several works have considered the problem of building tools for rapid creation of wrappers. One class of tools (e.g., [11]) is based on developing specialized grammars for specifying how the data is laid out in an HTML page, and therefore how to extract the required data. A second class of techniques is based on developing inductive learning techniques for automatically learning a wrapper. Using these algorithms, we provide the system with a set of HTML pages where the data in the page is labeled. The algorithm uses the labeled examples to automatically output a grammar by which the data can be extracted from subsequent pages. Naturally, the more examples we give the system, the more accurate the resulting grammar can be, and the challenge is to discover wrapper languages that can be learned with a small number of examples.

4.7 Matching objects

One of the hardest problems in answering queries over a multitude of sources is deciding that two objects mentioned in two different sources refer to the same entity in the world. This problem arises because each source employs its own naming conventions and shorthands. Most systems deal with this problem using domain specific heuristics.

5. Conclusions

In recent years, there has been a dramatic growth in the number of publicly accessible digital libraries on the Internet and all indicators suggest that this growth should continue in the years to come. Unfortunately, retrieving information from these digital libraries is not easy for several reasons. The first complication is distribution. Not every query can be answered by the data in a single digital library.

A second complication is heterogeneity. This heterogeneity may be notational or conceptual. Notational heterogeneity concerns access languages and protocols. This sort of heterogeneity can usually be handled through commercial products (such as the Sybase OpenServer). However, even if we assume that all digital libraries use a standard language and protocol, there can still be conceptual heterogeneity, i.e., differences in the relational schema and vocabulary. Distinct digital libraries may use different words to refer to the same concept and/or they may use the same word to refer to different concepts.

Mediation is a technology which inserts intelligent processing modules, called mediators, between servers and clients to provide value–added processing. As more implementations enter practice, the infrastructure grows and we expect that mediators can be installed rapidly and be maintained by their owners. Still there are specific issues that need to be taken care of in digital libraries to efficiently synthesize a response from multiple, heterogeneous information sources to a given query in a manner that is transparent to the user.

6. References

[1] Adali, S., 1996. Query Processing in Heterogeneous Mediated Systems. Ph.D. thesis, University of Maryland, College Park.

[2] Adam, N. R., Atluri, V., and Adiwijaya, I., 2000. SI in Digital Libraries, Communications of the ACM, vol. 43, no. 6, pages 64-72.

[3] Bell, D., and Grimson, J., 1994. Distributed Database Systems. International Computer Science Series, Addison Wesley, Wokingham, England.

[4] Bonnet, P., and Bressan, S., 1997. Extraction and Integration of Data from Semistructured Documents into Business Application. Proceedings of the International Conference on Applications of Prolog (INAP′97).

[5] Bonnet, P., and Tomasic, A., 1998. Partial Answers for Unavailable Data Sources. Proceedings of the 1998 Workshop on Flexible Query Answering Systems (FQAS′98), Department of Computer Science, Roskilde University. pages 43–54.

[6] Bressan, S., and Ibrahim, I. K., 1999. Semantic Query Transformation for the Integration of Autonomous Information Sources. Proceedings of the 12th Int1. Conference on Application of Prolog (INAP′99).

[7] Chaudhuri, S., Krishnamurthy, R., Potamianos, S., and Shim, K., 1995. Optimizing Queries with Materialized Views. Proceedings of International Conference on Data Engineering (ICDE), Taipei, Taiwan.

[8] Elmagarmid, A., Rusinkiewicz, M., and Sheth, A., 1999. Management of Heterogeneous and Autonomous Database Systems. Morgan Kaufman, Los Altos, CA.

[9] Florescu, D., Levy, A., and Mendelzon, A., 1998. Database Techniques for the World Wide Web: A survey. SIGMOD Record, vol. 27, no. 3, pages 59–74.

[10] Friedman, M., Levy, A., and Millstein, T., 1999. Navigational Plans for Data Integration. Proceedings of the National Conference on Artificial Intelligence.

[11] Haas, L., Kossmann, D., Wimmers, E., and Yang, J., 1997. Optimizing Queries Across Diverse Data Sources. Proceedings of the International Conference on Very Large Data Bases (VLDB), Athens, Greece.

[12] Hammer, J., and McLeod, D., 1993. An approach to Resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous Database Systems. International Journal of Intelligent and Cooperative Information Systems, vol. 2, pages51–83.

[13] Khalil-Ibrahim I., 2001. Semantic Query Transformation for the Intelligent Integration of Information Sources. Ph.D. thesis. Gadgah Mada University, Indonesia.

[14] Levy , A. Y., 2000. Logic–Based Techniques in Data Integration. In Minker, J., Logic Based Artificial Intelligence, Kluwer Publishers.

[15] Litwin, W., and Abdellatif, A., 1986. Multidatabase Interoperability. IEEE Computer, vol. 19, no. 12, pages10–18.

[16] Navathe, S. B., Elmasri, R., and Larson, J., 1986. Integrating User Views in Database Design. IEEE Computer, pages 50–62.

[17] Reddy, M. P., Prasad, B. E., Reddy, P.G., and Gupta, A., 1994. A Methodology for Integration of Heterogeneous Databases. IEEE Transactions on Knowledge and Data Engineering, vol. 6, no. 6, pages920–933.

[18] Silberschatz, A., Stonebraker, M., and Ullman, J. D., 1991. Database Systems: Achievements and Opportunities. SIGMOD RECORD, vol. 19, no. 4, pages 6–22.

[19] Tsatalos, O. G., Solomon, M. H., and Ioannidis, Y. E., 1994. The GMAP: A Versatile Tool for Physical Data Independence. Proceedings of the International Conference on Very Large Data Bases (VLDB), Santiago, Chile, pages 367–378.

[20] Widom, J., 1995. Research Problems in Data Warehousing. Proceedings of the 4th Conference on Information and Knowledge Management (CIKM).

[21] Wiederhold, G., 1993. Intelligent Integration of Information. Proceedings of ACM SIGMOD, vol. 22, no. 2, pages 434–437.

[22] Yang, H. Z., and Larson, P. A., 1987. Query Transformation for PSJ Queries. Proceedings of the International Conference on Very Large Data Bases (VLDB), Brighton, England, pages 245–254.

[23] Zisman, A., 1998. Information Discovery for Interoperable Autonomous Database Systems. Ph.D. thesis. Department of Computing, Imperial College of Science, Technology, and Medicine, University of London.