Report of the Workshop on Semantic Heterogeneity and Interoperation in Multidatabase Systems*

Pam Drew
U S WEST

Roger King
University of Colorado

Dennis McLeod
University of Southern California

Marek Rusinkiewicz
University of Houston

Avi Silberschatz
University of Texas - Austin

Abstract

This report presents a review of the problems that were discussed during the Workshop on Semantic Heterogeneity and Interoperability in Multidatabase Systems. The workshop participants discussed the importance of interoperation in the U S WEST information processing environment and the progress that has been achieved in three major research areas: resolution of semantic heterogeneity among cooperating heterogeneous systems, transaction management in such environments, and software architecture for interoperation. The workshop provided researchers with necessary feedback from the industrial perspective and helped in identifying the major issues that need further research. The following problems concerning the applicability of the methods proposed for data processing in heterogeneous, autonomous information systems have been identified: (i) many of the assumptions made in the research community are too restrictive to make the results directly applicable to existing environments; (ii) performance ramifications of various heterogeneous architectures need to be understood; (iii) the prototype systems need to be put to the test with real data, schemas, and transaction streams to verify their utility.

1 Objectives and Organization of the Workshop

The 1992 Boulder Workshop on Semantic Heterogeneity and Interoperation in Multidatabase Systems, sponsored by the NSF, DARPA and U S WEST Advanced Technologies, was third in a series started in Chicago in 1989 and continued in Tulsa in 1990. The main objectives of the 1992 workshop were:

- To provide a domain-specific forum for the exchange of ideas on the interoperability of heterogeneous and autonomous database systems between academic and industrial researchers.
- To provide a representative overview of the "state of the art" in the area, including the identification of problems that need to be solved and proposed solutions to these problems.
- To examine problems facing software application developers in the telecommunications industry, the applicability of existing research solutions and directions for new research.

The workshop was attended by invited researchers from academia, industry and government. The discussion was motivated by the "real life" problems presented by the industrial sponsor, U S WEST, which provided the setting for the workshop. The program consisted of plenary sessions, discussions in working groups, demonstrations of prototype systems and a panel discussion. This report presents an overview of the problems encountered in applying heterogeneous database methodology in an information processing environment typical of a large telecommunications company.

The report is organized as follows. In section 2 we present the problems in achieving interoperability between heterogeneous information systems in the U S WEST application environment. In section 3 we summarize the discussion on semantic heterogeneity in an environment of heterogeneous database systems. In section 4 we discuss limitations of traditional transactions applied in heterogeneous database systems. We present some approaches to relaxing properties of traditional transactions that may bring better performance

*The wokshop was sponsored, in part, by U S WEST, DARPA and NSF under grant IRI-9208743. However, the opinions, findings, conclusions and recommendations expressed in this report are those of the co-authors and do not necessarily reflect the views of the National Science Foundation, DARPA, or U S WEST. This document is available in electronic form by anonymous ftp from `ftp.cs.uh.edu` under `/pub/tech_reports/1992`.

while preserving integrity constraints. In section 5 we present alternative approaches to the integration of existing heterogeneous information systems in the context of the U S WEST application environment. Finally, in section 6 we summarize the results of the workshop.

2 Interoperation issues in the U S WEST information processing environment

The consensus among the participants, both from industry and from academia, was that the basic conditions for *interoperability* between heterogeneous computing systems (file transfer, remote login, etc.) either have been achieved or will be achieved in the near future. However, the system-level interoperability is not by itself sufficient to allow the development of complex applications spanning the boundaries of multiple systems that were designed to operate in a stand-alone mode. Rather, the achievement of interoperability should be viewed as an enabling condition for *interoperation* between application systems and semantic integration of information from diverse sources.

Interoperability among existing information systems requires resolving heterogeneities between computer systems as well as between data management systems. Achieving interoperation among existing systems requires resolving semantic heterogeneities in the meaning of data, data model heterogeneities and transaction management dissimilarities.

The current application environment of U S WEST, which is believed to be typical of large organizations with a broad scope of advanced data processing, can be viewed as a collection of heterogeneous information systems that are required to interoperate. Heterogeneity is one of the most dominant characteristics of U S WEST's base of software and database systems. This operational infrastructure was developed and extended over the past 25-30 years, and includes technologies ranging from assembler to C, and from sequential files to the occasional relational Database management Systems (DBMSs) – all running on different computing architectures and platforms. To further complicate matters, many of these systems were developed independently to support the three regional companies which now comprise U S WEST. This infrastructure can add to the operational costs of the company and make vital corporate information difficult to access, therefore approaches to streamline and rearchitect it are being investigated.

The streamlining efforts are focused on improving the way in which data is managed and processed. In U S WEST's traditional data processing environment, each system was built to automate some business function. Data and processing are tightly coupled and each application controls its own data. Data shared between systems is frequently passed through a large stream of applications. Each application in the stream transforms the data into a format that it can process, or a format that its internal data store can accept, before it actually performs its application function. The data is then passed on to the next system in the stream, frequently without being changed back into the format in which it was originally received.

This computing paradigm, complicated by a history of adding new functionality to existing systems, can lead to errors and wasteful computing. The errors include assignment of incorrect data to correct schema definitions, maintenance of inconsistent schema and data between replicated databases, inability to update incorrect and obsolete schema and data, and unnecessary processing of data.

Several steps must be taken to improve the quality and accessibility of the information in these systems. The first provides a mechanism for maintaining data consistency between existing databases that contain related data. This step will help correct inconsistencies that exist in the current information processing base. Second, integrated access to these heterogeneous, autonomous systems is provided so that applications requiring concurrent access to data currently maintained in separate information systems. can be constructed. The third step involves the creation of a separate collection of shared corporate subject databases that provide a conceptual single source of data and enforce the integrity constraints surrounding the data against the accessing applications. These new databases capture information required to support both existing and anticipated products and services. They will be populated with information from various sources including data in the existing application processing streams. Hence, the first step of correcting inconsistencies in existing data and the second step of providing integrated access to the existing data are required to support the third step of transferring the existing data to a new shared data architecture. The third step also requires the ability to keep the new databases consistent with any preexisting databases.

The primary topic of the workshop was the role of heterogeneous database approaches and technologies in these three activities. The discussions focused on the following issues related to interoperation among existing database systems and support for new applications that require an integrated access to the existing systems.

- approaches that capture the consistency requirements between existing systems; approaches that capture the semantics and integrity constraints currently embedded within applications that may also need to be integrated for access by other applications; and the criteria to choose a particular heterogeneous integration approach (i.e. multidatabases vs. global schema integration vs. some other interoperation model) in the U S WEST environment.
- the type of transaction management required to interact with the systems in the embedded base and to maintain consistency between replicated databases.
- feasibility of defining a shared data architecture and the technical approaches to its implementation.

In the following sections we present summaries and results of the discussions devoted to each of the above issues.

3 Semantic Heterogeneity

The context of our examination of semantic heterogeneity is an environment of heterogeneous database systems ("components") that are to an extent autonomous, and among which the sharing and exchange of information is desired.

Several types of heterogeneities among the components can be identified:

1. conceptual database model (meta-data specification technique/language),
2. ontological/terminological (application-specific abstractions),
3. conceptual schema (meta-data specification),
4. database content,
5. data format,
6. tool (database management system).

We consider levels 2 through 5 of inter-component variations to be *semantic heterogeneity.* It can be observed that a common technique or language is necessary to describe variations among information (level 1). We also note that tool heterogeneity is somewhat orthogonal to inter-component semantic differences (level 6). A candidate definition of semantic heterogeneity is "variations among component database systems in the structure, organization, and conceptual description of information facts (units), units of behavior (procedures), and semantic integrity constraints."

A key aspect of identifying and resolving semantic heterogeneity among a collection of components involves integrating (portions of) the content of those components. We note that this integration may involve meta-data (structures, units of behavior, and semantic integrity constraints) as well as specific data facts. There are several settings in which we can consider the semantic heterogeneity problem:

1. The collection of components is completely integrated into a single global database. Users and applications associated with each component must modify their functionality to utilize global integrated databases, possibly enhanced with tailored external views.
2. A "virtual" database is defined, consisting of parts of each of the components integrated into one global database. Users and applications continue to access their local databases, and utilize the global virtual database for external information.
3. A "federation" of components is formed, in which portions of each component are integrated into a specification which describes inter-component relationships. A centralized mediator for information sharing and exchange can be utilized, or the sharing and exchange can be implemented via direct connections among the components.
4. The components can interoperate on a pairwise basis, exchanging portions of their local information with other components.

We observe that in the above list, component autonomy is lowest in setting 1 and highest in setting 4. This comes at the price of increased chaos, which is highest in setting 4. The ability of users and applications to utilize (existing) local database tools and software is also highest in settings 3 and 4. In each of these settings, techniques are required to identify inter-component semantic heterogeneity, and to specify the relationships among the units of information in various components.

In the U S WEST application environment, there are roughly one thousand component systems. Each of these systems is itself complex, and a clear specification of the meta-data and specific data associated with a component is not available. Currently, U S WEST is using natural language descriptions to define business rules and to document existing applications.

Consider the goal of integrating selected subsets of these component systems. It can be observed that each component's portion may be dynamic with time, and that the integration process itself may alter what is to be shared. The importance of the role of semantic integrity constraints as well as application-specific units of behavior in the integration process was noted. It is clear that techniques for database integration are relevant here, but the problem of scale complicates things considerably. Further, the fact that a clear specification of each component is not available makes it difficult to apply database integration techniques directly in this context.

A number of important specific aspects of semantic heterogeneity in the heterogeneous database context have been identified. They constitute issues to be fruitfully considered in research and development directions:

- Semantics is relative. The meaning of information may be embodied within a database model, a conceptual schema, application programs, and the minds of users. A key goal is to make such semantics explicit so that it can be used to semi-automatically resolve semantic heterogeneity.
- Differences, variations, and conflicts among components may be considered inconsistency or heterogeneity: this is relative to the application.
- It is difficult to identify and resolve all the semantic heterogeneity among components. A useful goal may be to employ something like the "80/20" rule, which involves trying to handle 80% of the possible cases with 20% of the mechanism required for a complete solution.
- The extraction of semantics from application programs is central to the practical identification and resolution of semantic heterogeneity. Much of the meaning of data in existing data-intensive systems is embedded in code; a prerequisite to resolving semantic differences is specifying them in a manner conducive to resolution.
- A common technique or language can be used to support addressing semantic heterogeneity; the common technique would provide conceptual database model homogeneity as a basis for information sharing and exchange. While the precise nature of this technique is still open, standardization efforts offer some potential in this regard. Object-based ("semantic" or kernel "object-oriented") techniques are potentially viable candidates for such a common technique.
- Techniques are required to support the re-use of knowledge gained in the semantic heterogeneity resolution process. This may be placed in a knowledge base or "semantic dictionary". This ontological or terminological information is application-specific. Explanation facilities to describe this knowledge so that it may be effectively utilized would prove quite useful.
- The dynamics of inter-component sharing is critical. Information to be shared will change with time, and sharing mechanisms must allow for this.
- Both structural and behavioral semantic heterogeneity are important, and may be present at the conceptual schema and database content levels. The identification and resolution of semantic heterogeneity among behavioral units in different component systems is essential to support the effective reuse and composition of procedures to produce new and/or modified applications. Somewhere between structural and behavioral sharing lie semantic integrity constraints, which have the potential for providing substantial useful information for the resolution process.

In sum, a number of issues related to the semantic heterogeneity in heterogeneous information processing systems have been discussed. It is clear that the problem of identifying and resolving semantic heterogeneity in a heterogeneous database environment is far from solved, and that the problem itself is still at the stage of being understood.

4 Transaction Management in Multidatabase Systems

The problem of transaction management in an MDBS is to develop appropriate software, on top of pre-existing local database management systems (DBMSs), allowing users to execute transactions that access and update data residing at multiple databases. Before we discuss the particular characteristics of the MDBS transaction management problem and the nature of the desired solution, we briefly discuss the existing solutions to the transaction management problem in conventional systems.

A transaction is a means by which a user accesses and updates information stored in the database. At any instant, various transactions that share the database may be executing concurrently. A database management system protects transactions from the ill effects of other concurrently executing transactions. To do so, it ensures the following properties of transactions.

- **Atomicity:** All the effects of committed transactions appear, and no effects of aborted transactions appear in the database.
- **Consistency:** Successful completion of a transaction, if executed from a consistent database state, results in a consistent database state.
- **Isolation:** A transaction does not see the partial effects of other concurrently executing transactions; that is, a transaction is executed in a virtual single-user mode.
- **Durability:** Effects of committed transactions persist subsequent malfunctions.

These properties are together called the ACID properties of transactions. There are numerous reasons why the ACID properties are suitable for conventional transaction processing environments. Some of the most important are simplicity of use, generality of solution, ease of implementation, and acceptable performance.

Unlike the case of traditional database management systems, in an MDBS environment the preservation of the ACID properties of transactions has proved to be a difficult task, mainly due to the desire to preserve certain *autonomy* constraints on the design of the transaction management algorithm. One such constraint is that no changes be made to the existing software of the local DBMSs. Making changes to the software in order to either modify the design of the existing transaction management algorithm or to provide new functionality is too expensive, since it requires a substantial software engineering effort to identify the relevant portions of the code that would need to be altered. Further, since it is possible that the pre-existing software is tailor-made for high performance of local applications, changing the software may result in substantial performance degradation.

Another constraint on the design of the MDBS software is that each local DBMS retain complete control over the execution of the various transactions at that local site. Thus, a local DBMS is free to abort a transaction executing at its site at any time during its execution. The preservation of autonomy constraints makes the task of ensuring both global serializability and the atomicity of transactions difficult.

Various local DBMSs may follow different concurrency control protocols (e.g. two-phase locking, timestamp ordering, etc.), and thus the existing solutions for ensuring global serializability in a distributed database system that assume that each local DBMS follows the same concurrency control protocol cannot be used in an MDBS environment. One way to ensure global serializability is to redirect both local and global transactions to a common interface and to control the execution of all transactions at the interface. This, however, will result in a serious performance degradation since not only global but also local transactions will suffer concurrency control overhead at both the MDBS and local levels. The alternative is to control only the execution order of the global transaction operations. In this case, since the local DBMS may not make information about the relative serialization order of transactions known to the MDBS, the MDBS will need to make a pessimistic assumption that any two global transactions that execute at some common site conflict. This, however, will result in low concurrency and thus poor performance.

The problem in ensuring the atomicity of a global transaction arises because of the requirement that each local DBMS maintain complete control over transactions executing at its site. Since a global transaction executes at multiple sites, the atomicity requirement dictates either that all its subtransactions commit, or that all its subtransactions abort. To ensure this behavior, there is a need for an atomic commit protocol (e.g., two-phase commit protocol). An atomic commit protocol, however, requires each participating local DBMS to support a *prepared* state for transactions, in which it relinquishes the right to abort the transaction unilaterally. Since in an MDBS most participating DBMSs are pre-existing and were designed to be standalone systems, they may not provide for a prepared-to-commit state for transactions. Making changes to the recovery procedures of local DBMSs to support a prepared state is infeasible, since it would require substantial changes to the local DBMS software. The alternative is to *simulate* a prepared state at each site. To do so, both the local and global transactions would need to be redirected through the common interface, which, as argued above is not feasible for performance reasons. Even if each local DBMS did support a prepared state, the main objection to using an atomic commit protocol in an MDBS environment is that it results in a local DBMS's losing control over the transactions executing at its site (since a local DBMS can neither abort nor commit a global transaction in the prepared state). Thus, the only alternative may be to restrict the read and write accesses of global transactions in such a way that the atomicity of transactions can be ensured without the use of an atomic commit protocol. These restrictions on the generality of global transactions make the solution unsuitable for many MDBS environments.

Because of the above reasons, the participants in the workshop were in agreement that there is a need to develop techniques by which the semantics of the transaction and the database can be used to relax the ACID requirement of transactions. Two complementary methods for this were discussed. The first is based on exploiting the knowledge of database consistency constraints, while the second is based on exploiting the semantics of transactions.

With each database there is associated a set of integrity constraints. A database state is consistent if it satisfies the integrity constraints of the system. Thus, if *all* the integrity constraints of the system could be explicitly specified, then correctness could be ensured, without resorting to serializability, by the continuous monitoring of each transaction to determine that it see only consistent database states and that its execution not result in a violation of database consistency. There are, however, numerous problems with this approach. A system that continuously checks if database consistency is preserved will have poor performance. Depending upon the integrity constraints the above task may not even be computable. The ultimate argument against this approach is that, in practice, not all the integrity constraints of the database can be explicitly specified.

In contrast, it is possible that the *structure* of the integrity constraints of a database is known. In this case, it may be feasible to use such knowledge for the purpose of concurrency control. To illustrate this, suppose that one could partition the set of data items in a database into several disjoint sets such that there are no integrity constraints between the data items belonging to different partitions. However, arbitrary integrity constraints between data items within a partition are possible. In such a case, the integrity constraints of the system can be preserved by ensuring serializability within each partition. The above simple scenario can be generalized by permitting certain integrity constraints between the data items located in different partitions. If such constraints are relatively simple (e.g., copy constraint) and are explicitly known, then it may still be possible to ensure correctness by ensuring serializability within each partition and by using special techniques to deal with inter-partition constraints.

The above method for exploiting the structure of the integrity constraints is of interest from the MDBS perspective, since in an MDBS it may be feasible to assume that the integrity constraints between data items located at different sites are either non-existent or are relatively simple and can be explicitly specified. Some of U S WEST's applications that were discussed during the workshop exhibited this property.

Some research based on exploiting the structure of the integrity constraints to enhance concurrency in MDBS environments does exist. There are, however, many challenging research issues that need to be further addressed before the applicability of this approach can be determined. First, we need to examine if it is indeed possible to partition the set of data items in an MDBS in the manner suggested above. Then we need to determine the nature of integrity constraints that may be present between data items located in different partitions and further develop techniques that can be used to preserve the inter-partition constraints. Another research issue is to examine the relevance of the guarantee that such a system provides to the users. Consider, for example, a banking scheme where two accounts belong to different partitions such that there is no integrity constraint that relates the two accounts. In such a case, if a transaction that transfers money from one account to the other executes concurrently with an audit transaction that reads both the accounts, then it is possible that the audit transaction sees a sum that differs from the true balance of the two accounts. While this may be acceptable for certain instances, it may not be acceptable in general. One way to deal with this is to support different types of transactions in the system that explicitly specify their consistency requirements to the DBMS. For example, if the audit transaction needs to see the true sum of the balances in the accounts, it specifies such a requirement to the DBMS. This approach to relaxing the ACID requirements of transactions, where the basic idea is to exploit the transactions' semantics, was also discussed in the workshop.

A transaction specifies to the DBMS its consistency requirements, i.e. the degree of atomicity, isolation, consistency and durability that it needs for its computation to be meaningful. The DBMS ensures that the consistency requirements of each transaction are satisfied. For example, a statistical transaction may specify that it can tolerate up to one percent error in the value of data items it reads. Similarly, in a banking application a transfer transaction may specify that it can be interleaved arbitrarily with other concurrently executing transfer transactions. On the other hand, a transaction that computes monthly interest must be executed in isolation from transfer transactions for it to be able to compute the correct interest for each account.

In order to develop schemes that exploit the transaction semantics, there is a need to develop a new transaction model, with which a user can specify the exact consistency requirements of a transaction to the DBMS. However, before this can be attempted, there is a need to formalize what are the different degrees of atomicity, consistency, isolation and durability that a transaction may require for its successful computation. Clearly, there is a wide spectrum of properties, ranging from complete ACID preservation to the complete relaxation of the ACID requirement, that may suffice for correct execution of a transaction. There is a need to identify a subset of such properties that can be efficiently supported by the DBMS. Finally, there is a need to examine how such an approach can be used in multidatabase environments to overcome the autonomy-induced problems. One problem that seems incumbent in this approach is that the proper choice of the set of such properties that need to be supported may be application dependent. As a result, there may not exist a single transaction model that is suitable over all application domains.

Some research on exploiting the transaction semantics for transaction management exists in the literature. An example is the Saga transaction model in which the structure of the Saga and the associated compensating transactions specify the atomicity requirements of the transaction to the DBMS. Another example is the transaction model in which *breakpoints* within a transaction and the associated *compatibility sets* are used to specify the isolation requirements of transactions. Other related research efforts include the ConTract transaction model and the flexible transaction model.

There was general agreement among the participants in the workshop that while each of these research efforts is a step in the right direction and it is worthwhile to examine their applicability for transaction processing in multidatabase systems, there is a need to continue research to better understand how transaction semantics can be exploited for transaction management purposes.

5 Architecture

The discussion on the architecture of systems managing and processing shared data concentrated on the proposal to organize data accessed by U S WEST heterogeneous information processing systems into Corporate Subject Databases. The architecture of a target information processing system with shared data stored in Corporate Subject Databases was presented by U S WEST representatives. It assumes that the data will be extracted from the existing information processing streams and used to create a collection of shared, Corporate Subject Databases that can be accessed by new applications. There are several reasons for replacing the existing systems, including their high maintenance cost, inability to support new service offerings, high development costs due to their complexity, and the number of interconnections between those systems.

In the current U S WEST application environment, each application controls its own data and shared data is passed between heterogeneous systems through tailored, fragile interfaces. Frequently the same data is replicated in the data repositories of different systems and may have different format and meaning in different systems. Moreover, the consistency constraints may be embedded in the application code and since data updates are often not propagated between heterogeneous systems the data become inconsistent, e.g. the same customer of the telephone company may appear in databases of different systems as having different addresses.

In the Corporate Subject Databases architecture the data would be identified and modeled into subjects. The idea would be to isolate in one database all the information related to a particular subject; e.g. all data about the customers would be stored in a Customer database. Each subject database X would contain the unique master copy of its data, and all other copies of this data in the system would be derived directly from the subject database X. Then all the applications in the system would be gradually rewritten to manipulate data in these subject databases. Thus, instead of a number of vertically-partitioned systems (each with its own database and its own applications software), the new system would be horizontally-partitioned; i.e., it would consist of a number of applications all running on top of the suite of databases storing all information related to particular subjects. The subject databases would enforce the integrity constraints and provide a single source of data for applications.

Some participants of a discussion noted that the proposed architecture has a strict "subject" orientation that may result in the rigidity of the new system and in a tendency for the integrated system to perhaps not evolve gracefully. It was also pointed out that "subject" does not have a precise definition. Is Billing a subject, consisting of a set of related data, or is it a process? In general, it seemed arbitrary as to how the various subjects were chosen. Furthermore, they would naturally evolve over time, and thus the new system would someday need reorganization.

In contrast, other participants argued that the horizontal architecture allows all semantic integrity constraints to be defined and managed in a single Corporate Subject Database, thereby greatly simplifying and expediting their definition, retrieval, modification, and enforcement.

Some participants of the discussion noted that in general it was difficult to agree on terminology. What is a "subject"? What is a "function"? What is a "service"? What is a "skill"? It is hard to discuss the pros and cons of a subject-oriented versus a function- or service-oriented approach when the meanings of these terms are not well understood. However, other participants pointed out that such disciplines as Information Resource Management and Information Engineering have developed methodologies for defining and distinguishing "function" from "subject". Therefore, it seems necessary to work together with researchers in these fields to develop operational definitions of these terms.

The relative expense of maintaining the new system was also discussed. It was pointed out that the current vertical systems are already difficult to maintain and that the cost of maintenance would be much higher during the transition to the proposed new architecture. What would be the ratio of the time needed to rebuild to the time that the rebuilt system would be in use? In other words, is the effort to redo the system worth it? With respect to evolution, it was also noted that the data, the business processes, the corporate organization, and the technological environment of U S WEST will all change. Some of the participants shared the viewpoint that the new system would not adapt gracefully to these changes.

On the contrary, many industrial participants argued that the proposed architecture addressed the issue of simplifying the existing big and increasingly complex systems. This is because small modular building blocks with much simpler interface abstractions are fundamental features of the horizontal architecture.

During the discussion some of the participants suggested alternative organization of the U S WEST application environment. Perhaps the dependencies between the various vertical databases could be mapped. Then, this information could be used to keep the various databases consistent, with respect to how they are interpreted. New applications could be built directly on top of this vertical layer of coordinated databases, without ever having to materialize individual subject databases. These new applications could be built out of reusable services, in a mega-programming fashion. However, other participants argued that the above approach would increase the burdens of complexity, time, and cost of maintaining the existing systems, since a change to an existing system might result in an update to the dependency mappings.

The second major area of discussion concerned research that must be performed in order to understand the interoperation problem fully. It was noted that Wittgenstein said long ago that if an idea was worth discussing, then it was worth having a language to discuss it. Thus, researchers need to develop languages for handling data interoperation. These languages should contain primitives for pruning, restructuring, and merging data and schemas, for dealing with consistency and uncertainty, and for managing directories of meta-information.

A fundamental question concerned the nature of the information being manipulated by these languages. Should the atoms of these languages be data entities, attributes, services, and the like? Will these atoms change over time? If so, the languages must be extensible. The more general issue of information sharing was discussed. Perhaps one of the most difficult problems in developing integrated systems is in figuring out what schemas, data, and application code mean. Thus, standards for defining the meaning of software are needed.

Another major area of research that was discussed concerned active databases. It was suggested that such technology could be used to solve the interoperation problem partially. Active database constructs could be used to synchronize data updates, knowledge and data, and schema updates. Further, intelligent "mediators" could be used to act as intermediary agents between a vertical layer of databases and the application programs above them. This would help resolve mismatches in ontologies.

In sum, the discussion on the architecture of systems managing and processing shared data focused on U S WEST's plan for horizontally integrating heterogeneous systems, and novel research areas that must be pursued in order to achieve heterogeneous interoperation. The U S WEST plan to modify its data and system architecture was found highly interesting and generated considerable discussion. The discussion helped in developing significant insight into the data processing issues of a large telecommunications organization.

6 Summary

This report presents a review of the problems that were discussed during the Workshop on Semantic Heterogeneity and Interoperability in Multidatabase Systems. The workshop participants discussed the importance of the interoperation in the U S WEST information processing environment and the progress that has been achieved in three major research areas: software architecture for interoperation, resolution of semantic heterogeneity among cooperating heterogeneous systems, and transaction management in such environments. The results of the workshop can be viewed in two ways: from a perspective of industrial participants and from a perspective of the research community.

The industrial perspective in the area of integration of heterogeneous information systems was presented in all major areas of discussion during the workshop. The problems of data managing and processing in a large telecommunication organization, such as U S WEST, inspired the discussions and provided a context in which solutions proposed by researchers could be considered. Many industrial participants pointed out the following problems concerning the applicability of the methods proposed by the research community:

- while it is very satisfying to see several demonstration systems of varying degrees of robustness, these systems need to be put to the test with real data, schemas, and transaction streams to verify their utility;
- many of the assumptions made in the research community, such as integrating relational models only, are too restrictive to make the results applicable to the U S WEST environment, or any other information systems environment that is more than ten years old;
- performance ramifications of various heterogeneous architectures need to be understood, especially with respect to the different classes of query and transaction processing streams that could be found, and foreseen, in the U S WEST environment.

Finally, it was noted that heterogeneous database technologies can be used to support a wide variety of activities besides simple access to heterogeneous data repositories. These applications, such as migrating to a new data architecture or maintaining consistency between heterogeneous databases, may provide new requirements for heterogeneous database research.

From the industrial perspective, the workshop was considered to be a unique and valuable exchange of ideas. It was an opportunity to share the reality of today's information systems environments with the research community and to learn what new solutions may be on the horizon. The topic of heterogeneous databases is an applied area of research; interactions between industry, where the problem domain exists, and the research community, where potential solutions are investigated, are critical to making progress in the field.

From the perspective of the research community, discussions during the workshop helped in developing a significant insight into the data processing issues of a large telecommunications organization. They were useful to validate new approaches and technologies in an existing heterogeneous database environment. They also provided a look at the problems with data processing in heterogeneous information systems from the industrial point of view. The discussions during the workshop gave researchers necessary feedback from the industrial environment and helped in identifying the major issues that need further research. In our opinion they will help provide a further impetus for research in the area of interoperability and interoperation of heterogeneous information systems.

The consensus among the industrial and academic participants was that the formula of the workshop, namely concentration upon the specific problems of a particular industry, provided a unique opportunity for the exchange of ideas between the two communities, and that similar activities should be continue to be held, possibly including focus on other application areas.

Additional Reading

Proceedings of the First International Workshop on Interoperability in Multidatabase Systems, IEEE CS Press, Kyoto, Japan, April 1991.

Proceedings of the Second International Workshop on Interoperability in Multidatabase Systems, IEEE CS Press, Vienna, Austria, April 1993

Acknowledgments

Significant effort put forward by all of the participants and organizers contributed the workshop's success. In particular, we gratefully thank Yuri Breibart (Univ. of Kentucky), Meichun Hsu (DEC), Erich Neuhold (GMD), Hector Garcia-Molina (Stanford), Hans Schek (ETHZ), Richard Hull (USC), Witold Litwin (Univ. of Paris), Amit Sheth (Bellcore) and Gio Wiederhold (DARPA and Stanford) for their efforts chairing various sessions or serving on panels; Steve Melville (U S WEST), John Boughner (U S WEST), and Jim Dimino (U S WEST) for their presentations of the U S WEST environment; and Phil Cannata (MCC), Mansoor Ansari (Univ. of Houston), Linda Ness (Bellcore), Sudha Ram (Univ. of Arizona), Rich Corbett (U S WEST), Steve Laufmann (U S WEST), Randy Sparks (U S WEST), Rodolphe Nassif (U S WEST), and Nick Roussopoulos (Univ. of Maryland) for their prototype demonstrations. We also thank DARPA, the National Science Foundation, and U S WEST Advanced Technologies for financial support. Finally, we offer a special thanks to Ms. Sharon DeBaere (U S WEST) for her excellent administrative assistance.

Workshop Participants

Mansoor Ansari (University of Houston), John Boughner (U S WEST), Todd Boyer (U S WEST Communications), Yuri Breitbart (University of Kentucky), Phil Canatta (MCC), Wesley Chu (University of California), Robert Conway (U S WEST Communications), Richard Corbett (U S WEST Advanced Technologies), Gary Craig (U S WEST Communications), John Dakin (U S WEST Advanced Technologies), Jeff Dement (U S WEST Advanced Technologies), Jim Dimino (U S WEST Communications), Arvinde Donde (U S WEST Advanced Technologies), Pamela Drew (U S WEST Advanced Technologies), Jacques Durand (U S WEST Advanced Technologies), Martha Escobar (University of Southern California), Peter Fankhauser (GMD), Jean Claude Franchitti (University of Colorado), Hector Garcia-Molina (Stanford University), Pankaj Goyal (U S WEST Advanced Technologies), Sandy Heiler (GTE Labs, Inc.), Dennis Heimbinger (University of Colorado), William Hovey (U S WEST Advanced Technologies), Meichun Hsu (Digital Equipment Corporation), Richard Hull (University of Southern California), Mark Johnson (U S WEST Communications), Leonid Kalinichenko (Russian Academy of Sciences), Mohamed Ketabchi (Santa Clara University), Roger King (University of Colorado), Wolfgang Klas (International Computer Science Institute), Piotr Krychniak (University of Houston), Steve Laufmann (U S WEST Advanced Technologies), Barbara League (University of Colorado and DEC), Witold Litwin (University of Paris), Dennis Mcleod (University of Southern California), Sharad Mehrotra (University of Texas), Ken Mellberg (U S WEST Communications), Steve Melville (U S WEST Advanced Technologies), Rao Mikkilineni (U S WEST Advanced Technologies), Rodolphe Nassif (U S WEST Advanced Technologies), Linda Ness (Bellcore), Erich Neuhold (GMD), Calton Pu (Columbia University), Sudha Ram (University of Arizona), Rajeev Rastogi (University of Texas), Bruce Robinson (U S WEST Communications), Roger Rose (U S WEST Communications), Nicholas Roussopoulos (University of Maryland), Marek Rusinkiewicz (University of Houston), Hans Schek (Swiss Federal Institute of Technology), Ming Shan (Hewlett-Packard Laboratories), Amit Sheth (Bellcore), Avi Silberschatz (University of Texas), Margie Templeton (Data Integration, Inc.), Laural Thompson (U S WEST Communications), Ramesh Venkataraman (University of Arizona), Wally Wedel (U S WEST Communications), Gio Wiederhold (DARPA), Maria Zemankova (National Science Foundation), Jianhua Zhu (U S WEST Advanced Technologies).