UPDATES MANAGEMENT IN ENTERPRISES DATABASES

Yicheng Tao
Oracle Corporation
One Oracle Drive
Nashua, NH 03062, USA

Cheng Hsu
Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY 12180-359, USA

ABSTRACT

Managing concurrent updates in large-scale (distributed) enterprise databases is still an unsolved problem. There are many theoretical results reported in the literature, including the well-known two-phase locking method and its various revisions using, in particular, semantics-based markings to enhance the performance of the control mechanism. However, these results have yet been proven to be scalable for meaningful applications in significant distributed environments. To alleviate the problem, a new paradigm using the usage correctness criterion and the Metadatabase model to allow for flexible and adaptive control is proposed. The new work employs an architecture of concurrent rule-based control shells to effect a differential control mechanism supporting a range of integrity requirements for database updates management. The range supported includes both the traditional full control (instantaneous correctness criterion) design and the various relaxation designs utilizing user-specified prioritization control knowledge. A new global synchronizer is developed to enable the new approach, along with a control rule derivation and modification algorithm. The laboratory testing (LAN) shows that the new method, when implemented on the basis of the existing Metadatabase system, performs correctly as designed. Its theoretical properties are comparable to the traditional methods and yet its new capabilities are uniquely beneficial. An industrial testing (WAN) is also being conducted at Samsung Electronic Corporation for its Product Data Management system. The preliminary results reveal some highly promising improvement over the present technology used there.

1. AN OVERVIEW OF THE PROBLEM

Distributed database update is a hard problem when the number of concurrent transactions is non-trivial. A root cause to this problem is the complexity of serializability which achieves the correctness of updates synchronization and assures the consistency of databases. The complexity in its purest sense is a function of the number of (common) data items that exists in the distributed environment; however, in actuality, it is compounded many fold by practical computing issues at the network, the database management system, and the operating system levels. In a wide-area networked, heterogeneous environment the overall complexity is practically intractable. This is a basic reason why large organizations resort to data warehouses or other technologies even though they do not support real time and online database updates. A prime case showing this problem is the Product Data Management (PDM) system at the Samsung Electronic Corporation (SEC). This case is documented and analyzed in [13].

The complexity of serialization needs to be managed or even simplified so as to show a meaningful performance of updates for multiple databases and thereby close up the gap between the practical solutions which do not support updates and the theoretical ones which can not be implemented adequately. The question is, of course, how? The answer that we developed, and tested, in this research is to allow an *adaptive reduction of control* according to usage criterion and to *execute the control concurrently* at all local sites involved.

Traditionally, distributed database updates are transacted under the assumption that all data items must be controlled and consistent all the time at all places. The reality, however, shows that different applications tend to have different and time-varying demands on control for different local data of the HDDBMS (Heterogeneous Distributed Database Management System); some require instantaneous consistency and others only periodical updates. The use of replicate data to "localize" global updates in actuality is a testimony to the fact that not all data require instantaneous updates (data replication *per se* does not support any instantaneous consistency, of course). Thus, to accommodate all eventualities, the traditional control model, which is determined at the design time, must target at the maximum scope of possible global data with the minimum tolerance for their inconsistency. This is a root cause for the complexity of serializability.

This assumption poses serious problems: (1) performance problem, (2) adaptiveness problem, and (3) implementation problem. The performance problem is caused by conflicts resolution and network traffic. Typically, the serializability criterion of correctness is used to model and resolve the read/write conflicts in database transactions. A great deal of research have been conducted for resolving the global serializability problem. The results for concurrency control can be classified into two major categories: (1) serializability-based approach [2,5], and (2) serializability relaxation approach [3,4,11]. The first approach is the majority, but the second represents a need to circumvent the difficulties associated with serializability. The latter

tries to relax serialization as much as semantics of transactions can be defined *a priori*, and it is reduced to the first approach if no semantics information is available. Conflicts resolution using the first approach and the second approach in the worst case requires the maximum degree of participation of the HDDBMS at all time and the system would have no resource (or the user would have no way to prevent this from happening). Therefore, performance is impeached in terms of synchronization for distributed database updates.

The maximum participation of local sites requires maximum network communication. In addition, network traffic problem may also arise from an inflexible design of execution model. For example, the data replication technique (for either fault tolerance or performance improvement) needs to replicate only the necessary remote data, otherwise the network traffic would suffer from redundant replication processes and might actually lose whatever advantage it could gain. The definition of necessary replication cannot be determined once at the design time and remain adequate for all run time.

The adaptiveness problem is caused by the fact that the requirement of synchronization on every data item cannot be pre-defined and remain fixed throughout, as assumed in the previous approaches. Therefore, the "hard-coded" control model without any structure for online modification gives the inability to adapt the dynamic changes of requirements on the fly. This means that the serialization-based approach would not be able to add or delete any global data item after the scope of synchronization is determined (at the design time), and that the relaxation-based approach cannot modify its relaxation.

The third problem, implementation problem, is caused by the control requirements that interfere local transactions. In general, the more items that are controlled, the more local site interference in terms of I/O, CPU and other resources processing is required. Furthermore, serialization issues such as deadlock and network reliability become intractable in, especially, WAN and global network environments where overall network management is lacking and instantaneous correctness is physically impossible due to delay in communication [6]. Altogether, they comprise a complexity of implementation that increases rapidly as the scale of HDDBMS increases. Again, there is no built-in structure in the traditional approaches to alleviate this (scalability) problem, let alone preventing or solving it.

Consequently, the traditional approaches have not yet sufficiently solved the distributed databases update problem. The research problem we take on is how to alleviate the above problems concerning performance, adaptiveness, and implementation. Specifically, our objective is to provide reconfigurable concurrency control allowing the user/modeler to flexibly require synchronization of distributed database updates (according to the usage criterion of their applications) and thereby reduce the complexity of serialization and optimize the performance.

2. THE SOLUTION APPROACH

A solution approach that promises to reduce network traffic and minimize the need for global serialization has been proposed earlier in the Metadatabase model, using the ROPE (Rule-Oriented Programming Environment) methods [1,7,9] to provide differential control based on the usage correctness of data updates. Only the data that truly need instantaneous correctness are subject to serializability control and are updated according to their usage requirements in this approach. The upper bound of global control is determined and derived from the global information model consolidated at the metadatabase as update integrity rules. The control scheme (instantaneous vs. various classes of usage correctness) for these update rules is defined by the user/modeler as control rules and both are integrated and distributed to ROPE shells for concurrent processing. These differential control rules would then be executed by using the concurrent rule-based shells under the administration of the metadatabase (as the HDDBMS). This differential control approach, however, lacked a complete execution model prior to this research. It had not been fully implemented, nor verified. Previous methods did not, for instance, provide an instantaneous correctness mechanism that optimizes serializability by utilizing the ROPE capabilities, nor a scheme of semantic markings for defining usage correctness on data items to effect customized control on distributed database updates. It could provide only the "no control" (or the best-case) mode of the differential control concept in the previous results. The practical feasibility, scalability, and even relevance of this approach had not been established because this lack of an execution model for the concept.

We have in this research developed a differential global synchronizer (different from the traditional serializability approaches) to resolve the problem of distributed database updates. The differential control method synthesizes the appropriate results available in the traditional literature and incorporates them into the Metadatabase model to provide a promising solution which has been shown to be correct and scalable for practical conditions of distribution involving LAN and WAN. The new results capitalize on the rule-based structure to adaptively adjust the control requirements (ranging from no control to full control) and thereby alleviate the performance, adaptiveness, and implementation problems discussed above. The new method presents an improved design for serializability using concurrent processing over the previous two traditional approaches to serializability.

3. THE RESEARCH METHOD

The new results need to address the following basic challenges facing the differential control approach: (1) how to derive and determine integrity rules, (2) how to declare control knowledge, (3) how to distribute rules, (4) how to synchronize distributed database updates, (5) how to prove the correctness of the synchronizer and the new usage criterion, and (6) how to establish the scalability of the solution. In addition, they also address the fundamental challenges of how to provide a generic framework so that the logic of the differential control method will not be affected by the evolution of implementation technology such as JAVA, CORBA (Common Object Request Broker Architecture), or ATM (Asynchronous Transfer Mode); nor by the domains of application.

To meet the former (six) challenges, the research first employed the existing Metadatabase results which have responded to these challenges under the condition that global serialization is completely relaxed (or the best case scenario). On this basis, the research then focused on developing new methods that support additional levels of control using data management rules based on instantaneous correctness, and then integrated all results into a complete differential control algorithm. In particular, the following tasks pertaining to the new methods of differential concurrency control have been undertaken in this research: (1) determining a framework for deriving the upper bound of control - i.e., the global data integrity rules, (2) defining the semantic markings (control knowledge) for global data management requirements information, (3) incorporating these markings into rule syntax, (4) extending the structure of the ROPE shells under the new conditions, (5) developing the synchronization methods and algorithms, (6) developing the concurrent message processing methods and algorithms, and (7) constructing the overall differential control method and algorithm. These results are integrated with the previous Metadatabase methods to further enhance the design for multiple databases in both LAN and WAN environments. Finally, a formal analysis of the methods proving the correctness of the algorithms and assessing the overall performance of the approach is conducted. The methods and algorithms as well as their theoretical properties are presented and analyzed in [13].

In addition, to verify scalability and practical feasibility, the research developed a laboratory prototype to test the new design. Some scenarios taken from a Computer-Integrated Manufacturing (CIM) example are used to conduct some experiments on the prototype with respect to the goal of differential control of distributed database updates. The performance is analyzed and assessed against the traditional approaches. Furthermore, the new results are implemented in Samsung Electronic Corporation. The scalability from a laboratory environment to a significant, large scale industrial setting is satisfactorily illustrated [13].

To address the implementation challenges, we separate the generic design logic from its implementation techniques. To verify this point, we have shown that the same logic (i.e., the core ROPE structure) is implementable in three classes of technology: stand-alone PC (or mainframe/workstation), client-server, and PLC (Process Logic Control - real time process). The resulting versions are referred to as ROPE-{SYMBOL 97 \f "Symbol"}, ROPE-{SYMBOL 98 \f "Symbol"}, and ROPE-{SYMBOL 103 \f "Symbol"}, respectively; which are all compatible and integrated in the prototype. Moreover, in the prototype, we implement the differential control concept using portable standards (e.g., SQL3 and common C), which is adaptable to most situations. The aspects that are dependent on implementation are only the interface design and the global language used. The overall structure and processing logic will stay the same. For example, CORBA or ATM would alter the current implementation of the network communication components but would not change the prototype's logic of messaging protocol, nor supersede it. Similarly, should SQL3 be replaced by a more powerful language for rules and objects processing, only the coding of database access would be simplified but not the tasks of metadata nor that of the rule execution model. Thus, unless the new technology would provide the global information model and implement automatically the control scheme and the rule-based execution model, it could only affect the physical implementation of the research results, but not its logical validity. The generic nature of the differential control method also addresses the research issue of generalizability. Both the laboratory prototyping and the empirical results achieved at Samsung have provided promises in this regard.

4. THE RESULTS

There are six major results achieved in this research. Together, they constitute a complete differential control method and thereby result in a verified Metadatabase solution to the distributed database update problem. The results are summarized below.

(1) Usage-based correctness criterion. The research provides a new usage correctness criterion which selectively relaxes serializability on distributed database updates based on consistency requirements. A rule-based representation method is developed to implement this new criterion. The knowledge about the distributed databases and their applications (i.e., the information models, operating rules, and integrity rules) in the metadatabase is employed and deployed to build concurrent control shells around each application.

(2) Adaptive control model. Since the new usage criterion is built on concurrent rule-based shells, a

formal model is developed to represent and modify the database update control requirements at both the design time and the run time. This model allows control rules and integrity rules to be dynamically distributed and updated; therefore, it achieves adaptiveness that is unavailable in the literature until now.

(3) Concurrent messaging protocol. A new message processing method is developed to maximize the benefits of the concurrency design. Its efficiency is obtained from three techniques: client-server architecture, asynchronous messaging, and concurrent message/rule processing. The client-server architecture provides direct remote access within LAN, which allows for event-based communication among local servers. The asynchronous messaging supports WAN communication since the message sender does not need to wait for response from the message receiver. The concurrent message/rule processing is achieved by the *threading* technique supported by multi-tasking operating system. Altogether, the messaging protocol helps suppressing the total processing time required for given transactions and network traffics.

(4) Global integrity rule generation. The automation of data management/integrity rules determination and derivation gives the correct requirements of maximum control, as the upper bound of the differential control method. A total relaxation of all rules (through the above control model) gives the correct lower bound.

(5) Differential control algorithm/method. An overall serializability algorithm is developed to achieve instantaneous correctness as the traditional approaches do. However, it is designed on the basis of the above results and hence utilizes the rule-based capability to support flexible requirements of serializability. It also improves on concurrent processing of some tasks of the serializability control using the concurrent shells and messaging protocol.

(6) Analytical and empirical verification of the new method. This new algorithm is shown to be correct and at least as efficient as the traditional approaches when no flexibility is required (i.e., under the traditional assumption). The analysis is carried out with respective to the best case, worst case, and the expected case, against both of the traditional approaches (serialization and relaxation). The empirical testing is conducted in the laboratory and Samsung. The overall performance is reasonable and is consistent with the theoretical analysis.

5 THE COMPLETE SOLUTION

We now describe how these new results work together for distributed database updates using the Metadatabase as an HDDBMS.

The extended concurrent architecture that we propose to effect the new differential control method is depicted in Figure 1. This architecture includes both client-server and stand-alone systems in LAN, WAN, and global network environments. The metadatabase itself (a rigorously constructed collection of enterprise metadata) provides an integrated enterprise model for the multiple information systems it integrates. This model abstracts their databases and the interactions among the different systems; i.e. the information contents and their contextual knowledge. The metadatabase (1) uses the enterprise model to determine the update integrity rules (the upper bound), (2) assists end-users articulating the global control requirements (at both design time and run time) for distributed database updates, and (3) distributes the control knowledge as rules to ROPE shells. The shells in the architecture, therefore, implements the distributed (localized) knowledge which, in turn, is managed by the metadatabase. It is worthwhile to note that the representation method of metadatabase treats information models as metadata "tuples" of certain base tables in the metadatabase [10], thus it is able to incorporate legacy, new or changed local models into its generic structure of metadata to support evolution without system redesign or recompilation.

{EMBED CDraw \s * mergeformat}

Figure 1 : The Extended Concurrent Architecture Using a Metadatabase

In this architecture, two versions of ROPE shells are employed: (1) ROPE-{SYMBOL 97 \f "Symbol"}, for traditional database and file systems [1,8]; and (2) ROPE-{SYMBOL 98 \f "Symbol"}, for database servers using LAN and middleware [12]. The main reason for the difference is to respond to the two-version approach requirements and capabilities of the vastly different technologies. While ROPE-{SYMBOL 97 \f "Symbol"} is more general and generic, ROPE-{SYMBOL 98 \f "Symbol"} takes advantage of ANSI ISO/SQL3 definitions and direct remote access protocols among servers.

To facilitate WAN-based interoperations, the method of metadatabase-defined *data replication* is used as a mechanism for fault-tolerance. The basic objective of fault tolerance is to prevent the operation of local systems from being disrupted by the failure either at other systems (belonging to the same cluster or other WAN-/LAN-based clusters in the distributed environment) or at the network. This mechanism of select data replication is useful for enhancing data availabilities and can also be used to improve performance.

With the concurrent architecture, the Metadatabase model supports full concurrent processing among local

systems according to operating rules, and provides an alternative to relying entirely on global serialization for concurrency control of distributed database updates. The usage correctness criterion is defined as follows: a data value only needs to be updated when it is needed by an application according to its contextual knowledge. Therefore, the consistency is established on a need-to-know basis, using both operating and integrity rules (also referred to as data management rules) defined in the enterprise model. This criterion minimizes the need for a central controller, depending on the rules used. The rules can be tailored to differentially provide a variable level of consistency for different data items; while, in contrast, serialization presumes that all data be equally critical and need complete synchronization at any point in time. Moreover, the concurrent architecture allows for a serializability mechanism to be employed at some particular sites of the multiple systems. For data items that require instantaneous correctness, operating rules can be developed to impose synchronization on them with a serializability controller in a conventional (but improved) manner.

The global consistency as measured against the new criteria is therefore achieved in two stages: the correct modeling of the data management rules to represent the logical correctness, and the accurate implementation of these rules to ensure the logical correctness. Specifically, the first stage involves three levels of control for database updates: (1) no serializability control (independent concurrent processing), (2) relaxed control according to usage requirements, and (3) control with instantaneous correctness criterion. Every data item can be marked for any of these three levels of control and the marks can be changed at any time through rule-based updates. Any percentage of the collection of all data items can be put under any level of control. The first level of control is the default, which means no information for differential control is specified, hence the database updates will not be synchronized by a central serializability controller. It is also equivalent to the case of synchronization where all updates are compatible. The other two levels of control require specific information defined by the user/modeler regarding how data updates should be synchronized at any given point in time. The information for differential control consists of seven components: (1) <decl-rules>, a set of integrity rules that are declared to be controlled with the same information; (2) <sync-items>, a set of items to be synchronized by the controller for database updates; (3) <enforcing-time>, an indicator for the level of relaxation on instantaneous correctness that the <sync-items> must be synchronized; (4) <waiting-time>, the maximum waiting time for the global queries of rule execution; (5) <mirror-sites>, the alternative (replicated) data that can be used in rule execution when the waiting time for the global queries is over; (6) <priority>, the execution priority for <decl-rule>; and (7) <symmetric-rule>, the rule that might cause “lost update” and “cyclic update” for <decl-rule>. The second component is required when the user/modeler wants to declare instantaneous correctness on certain data items; others are optional (might be determined by the system).

With these information for differential control and the capabilities of ROPE shells, the second stage (i.e., execution) for the database updates synchronization is achieved in a distributed manner to further enhance the performance for the select data items that truly need instantaneous correctness. The synchronization proceeds as follows: when an update is initiated (at any site), a message is sent to the global synchronizer at the metadatabase server. The synchronizer then uses a *transaction structure* which is dynamically built for all active transactions that involve data items in <sync-items> to detect if there is any read/write conflict with the database updates. If there is no conflict detected, the synchronizer will re-build the transaction structure by adding the new transactions (i.e., the distributed database updates) and continue the synchronization work. Otherwise, the synchronizer will queue the request until the requested resource is available. The actual processing of data and rules in the update transactions takes place only at the local sites involved through ROPE shells. The processing is executed in exactly the same manner as the no-control transactions that are concurrent and independent, barring from the locking messages they process in addition.

In this approach, the differential control method is fully integrated with the previous Metadatabase model. Thus, the research both employs the particular elements of the metadatabase and ROPE shells to bring about the solution, and extends the previous model for multiple database integration and management.

6. CONCLUSIONS

The research has contributed to the literature a new distributed database update algorithm. This algorithm provides new abilities to (1) selectively reduce the complexity of serializability according to usage criterion, (2) adaptively modify the requirements of serializability, and (3) concurrently process the transactions of serializability. The first two properties add new basic capabilities to previous results when flexibility is either required or available; while the third represents an improvement on performance in all conditions.

Secondly, the determination and derivation of data management rules -including the traditional pure data update rules and the more general global integrity control rules - itself is a significant contribution. There have been no such integrity rules reported in the literature. With the (rule-based) differential control method, integrity control can be implemented on the same basis as the pure database update rules once they are determined.

The significance of the new results is demonstrated in a Samsung case study. In the Product Data Management (PDM) system at SEC, a distributed database update program dubbed as the ChonJi system was developed jointly by Samsung and Hewlett-Packard (HP) to expressly manage the creation of new parts among multi-servers, multi-divisions. In SEC's environment, HP's WorkManager is employed for workflow management. The main task for ChonJi is to periodically cascade updated parts data from one PDM database to another.

However, the sheer size of the table involved and the network traffics have plagued ChonJi and resulted in very poor performance. As an ongoing experiment, SEC has test-implemented the core design of ROPE-{SYMBOL 98 \f "Symbol"} (using ANSI ISO/SQL3 standard trigger/log definitions) to handle the same job. The case study shows impressive improvement over ChonJi, because of ROPE-{SYMBOL 98 \f "Symbol"}'s ability to perform differential updates - replicated sites, timing, items, and priority.

In addition to HDDBMS *per se*, the differential control method can also be applied to data replication rules for fault tolerance or complex business rules enforcement. Other applications such as distributed workflow process management, *insensitive* data transactions [4], distributed teamwork [11], and data forecasting are also areas for direct application.

7. REFERENCES

[1] Babin, G., *Adaptiveness in Information System Integration*, Unpublished PhD Thesis - Decision Science and Engineering Systems, Rensselaer Polytechnic Institute, Troy, N.Y., August 1993.

[2] Bernstein, P. A., V. Hadzilzcos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley, Reading, MA, 1987.

[3] Farrag, A. A. and M. T. Özsu, "Using semantic knowledge of transactions to increase concurrency," *ACM Trans. on Database Systems*, 14(4), pp. 503-525, December, 1989.

[4] Garcia-Molina, H., "Using semantic knowledge for transaction processing in a distributed database," *ACM Trans. on Database Systems*, 8(2), pp. 186-213, June 1983.

[5] Gray, J. N., *Notes on Database Operating Systems*, Unpublished lecture notes, IBM San Jose Research Laboratory, San Jose, CA, 1977.

[6] Holtkamp, B., "Preserving Autonomy in a Heterogeneous Multidatabase System," in *COMPSAC*, pp. 259-266, October 1988.

[7] Hsu, C., *Enterprise Integration and Modeling: the Metadatabase Approach*, Kluwer Academic Publishers, Boston, 1996.

[8] Hsu C. and G. Babin, "A Rule-Oriented Concurrent Architecture to Effect Adaptiveness for Integrated Manufacturing Enterprises," in *Int'l Conf. on Industrial Engineering and Production Management*, pp. 868-877, June 1993.

[9] Hsu, C., G. Babin, M. Bouziane, W. Cheung, L. Rattner, and L. Yee, "Metadatabase Modeling for Enterprise Information Integration," *Journal of Systems Integration*, 2(1), pp. 5-39, January 1992.

[10] Hsu, C., M. Bouziane, L. Rattner, and L. Yee, "Information Resources Management in Heterogeneous, Distributed Environments: A Metadatabase Approach," *IEEE Trans. on Software Engineering*, 17(6), pp. 604-625, June 1991.

[11] Lynch, N., "Multi-level atomicity," *ACM Trans. on Database Systems*, 8(4), pp. 484-502, December 1983.

[12] Tao, Y., C. Hsu, and G. Babin, "Flexible Integration of Manufacturing Database Using a Model-Based Architecture Capable of Wide-Area Networking," in *5th Int'l Conf. on Data and Knowledge Systems for Manufacturing and Engineering*, October, 1996.

[13] Tao, Y., *A Differential Control Method for Distributed Database Updates Using Rule-Based Shells*, Unpublished PhD Thesis - Decision Science and Engineering Systems, Rensselaer Polytechnic Institute, Troy, N.Y., August 1997.

Filename:	SMC.DOC
Directory:	F:\TAOY\THESIS
Template:	D:\MSOFFICE\WINWORD\NORMAL.DOT
Title:	
Subject:	
Author:	.
Keywords:	
Comments:	
Creation Date:	06/15/97 9:33 PM
Revision Number:	17
Last Saved On:	06/15/97 11:24 PM
Last Saved By:	.
Total Editing Time:	99 Minutes
Last Printed On:	12/05/97 12:53 PM
As of Last Complete Printing	
Number of Pages:	6
Number of Words:	4,617 (approx.)
Number of Characters:	26,322 (approx.)