Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 3, March 2015, pg.389 – 395

RESEARCH ARTICLE

Distributed Transaction Processing Model Based on Image Technique in Mobile Computing

Veenu Saini[1], Roshan Lal Hiranwal[2]

[1]Department of IT, PGGC, Chandigarh, India

[2]Department of Computer Science, PGGC, Karnal, India

[1] vnusaini@yahoo.co.in, [2] roshanhiranwal@gmail.com

Abstract— Advancements in networking and distributed processing are enabling the emergence of new types of distributed processing environments. The environment for accessing and processing information is rapidly changing from stationary to mobile and location independent. Mobile users are more likely to face with more disconnection because of the properties of the mobile environment. Consequently, transaction processing and efficient update techniques for mobile and disconnected operations have been very popular. Because the traditional techniques do not function properly in a disconnected distributed environment, new mechanisms are to be developed for the management of mobile transaction processing.

This paper thus focuses on a distributed transaction model that resolves issues of mobile computing environment such as disconnection transaction processing, distributed execution, mobility and transaction properties by using a new technique; we have referred this model as Distributed Transaction Processing Model and the technique used to implement this concept is Image Technique.

Keywords-- Transaction Models, Distributed Transaction, Mobile Computing, Transaction Processing, Mobility

I. INTRODUCTION

Mobile transaction is a transaction performed with at least one mobile host takes part in its execution; also, it may be defined with perspective of its structure as a set of relatively independent (component) transactions, which can interleave in any way with other mobile transactions. The mobile user, by nature, is moving from one place to another so the mobile transaction should follow the user anywhere, which is not supported in distributed database transactions. We view a transaction as a program in execution in which each write-set satisfies the ACID properties and the program that updates the database as a three folds module (phases) reading phase, editing phase, and validation and write phase.

II. MOBILITY CONSTRAINTS

Mobile computing [4] is characterized by four constraints:

A. Mobile elements are resource-poor relative to static elements.

For a given cost and level of technology, considerations of weight, power, size and ergonomics will exact a penalty in computational resources such as processor speed, memory size, an disk capacity. While mobile elements will improve in absolute ability, they will always be source-poor relative to static elements.

B. Mobility is inherently hazardous.

A Wall Street stockbroker is more likely to be mugged on the streets of Manhattan and has his laptop stolen than to have his workstation in a locked office be physically subverted. In addition to security concerns, portable computers are more vulnerable to loss or damage.

C. Mobile connectivity is highly variable in performance and reliability.

Some buildings may offer reliable, high-bandwidth wireless connectivity while others may only offer low-bandwidth connectivity. Outdoors, a mobile client may have to rely on a low-bandwidth wireless network with gaps in coverage.

D. Mobile elements rely on a finite energy source.

While battery technology will undoubtedly improve over time, the need to be sensitive to power consumption will not diminish. Concern for power consumption must span many levels of hardware and software to be fully effective.

These constraints are not artefacts of current technology, but are intrinsic to mobility. Together, they complicate the design of mobile information systems and require us to rethink traditional approaches to information access.

III.MOBILE TRANSACTION PROCESSING ISSUES

Mobile transactions are long-lived, bound to many different types of mobile devices, involved in heterogeneous database and network and execution time is varying.

This section focuses research challenges in mobile transaction [2][4][5] mainly on mobile database, service handoff and scheduling.

A. Mobile Database

Currently, the mobile transaction is developed on the top of currently existing database system. Most of mobile transaction models are based on the earlier discussed mobile environment. In this environment, the database resides, replicated and distributed on the fixed hosts in wired network. However, the capacity of mobile computing device is expanding and a MH can become a host for data processing or a place to store the native data. In this case, the physical location of database system is changing. Identify the location of the MHs which stores the required data is one of the major issues in mobile database [9]. To obtain optimization on query processing, databases are replicated or fragmented in MH. Because of the disconnection and mobility of MH, maintaining data consistency between MH is more complicated. Location dependent data [14] also needs to be considered.

B. Service Handoff

When a MH moves into a new region, a new BH is assigned to this MH. Information about current transaction state is saved and transferred from old BH to next BH. This operation sometimes is unnecessary because not all the time MH requires assistant.

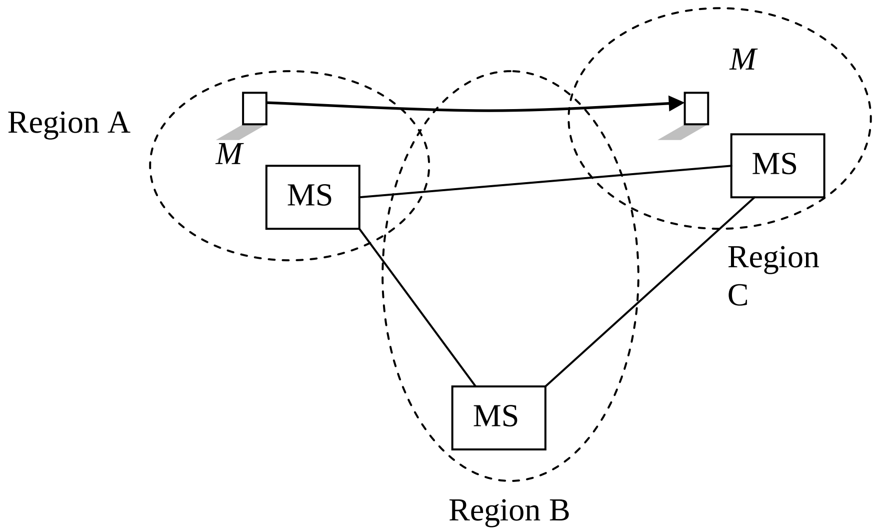


Figure 1 Service Handoff between BH

Figure 1 illustrates the situation. MH *M* is moving from region A to region C through region B. However, in region C the MH does not need any assistant from BH in region B. The information about transaction state should directly forward to BH in region C. This information package also includes the hardware profile of MH, context of application and environment. If this information is stored at MH then the MH can become an active element, which can initiate a connection when needed. The question is how a MH finds out what BH it should connect to. Currently, when a MH wants to exchange information with another MH then it has to rely on the support from at least one BH. How can one MH directly obtain communication channel with other MH?

C. Scheduling

Execution time of mobile transaction is varying. Mobile transaction can easily miss its required deadline due to its mobility and portability. It is not applicable in mobile transaction if a missing deadline transaction is always aborted. Missing deadline causes inconsistency in global state of transaction and blocks other transaction's execution. Enforcing technique like earliest-deadline-first [10] can be applied. Mobile transaction requires flexible scheduling mechanism. Scheduling a transaction in a FH is different from MH. Schedule in mobile transaction should take into account the mobility of MH in both location and time. MH should be able to reschedule its execution plan according to its physical state (power, communication bandwidth).

D. Caching

Caching of data at MHs can improve performance and facilitate disconnected operation. Much research has been performed in the area of MH caching. Caching issues are complicated by the use of Location dependent data (LDD). Because of the fact that data, which is cached, can be viewed as a temporal replica of spatial data, as a MH moves into new data regions the cached data may become obsolete. This data is not stale because it is incorrect, but may not be desired because it is from a foreign region. Replacement policies need to be re-examined to include location information. For example, data from a foreign region should perhaps be replaced before data from the current home region even though the foreign data is more recently used. However, this is further complicated by the fact that ongoing or future queries could be bound to foreign regions. The MH mobility is such that the MH could very quickly move back into the home region for this data, making the re-placement policy also subject to movement of the MH.

The need to maintain and manage a large amount of data efficiently has been the driving force for the emergence of database technology and E-commerce. Initially, data was stored and managed centrally. As organizations became decentralized, the number of locations, and thus local databases, increased. The need for shared access to multiple databases was inevitable. Geographical distribution of data, demand for highly available systems and autonomy coupled with economical issues, availability of low cost computers, advances in distributed computing, and the demands of supply chain based E-commerce, are among the pressing issues behind the transition towards distributed database technology. Design of distributed database management systems (DBMS) has had an impact on issues such as concurrency control, query processing/optimization and reliability.

IV. IMAGE TECHNIQUE

Most of the work handling mobile transactions as (Kangaroo, Reporting and Co, Moflex, Escrow techniques etc.) assume that the handoff process is under the mobile support station (MSS) responsibility, and the mobile support stations has the capability to transfer control and transaction history among servers. However, this approach has many limitations, such as, if the mobile unit moves relatively slow such that the probability of the commitment protocol terminating at the same cell is high. If it is fast moving then a frequent migration of the control may increase the protocol latency and thus its vulnerability. In addition, if a big number of MUs move among cells, so that most of the response time is spent in transferring data among cells.

Most of the used methods apply the concept of compensation. A compensating transaction is a transaction with the opposite effect of an already committed transaction. It is intended to undo the visible effects of a previously committed transaction, e.g., cancel car is the compensating transaction for rent car. A problem lies in the fact that compensation does not reserve database consistency: for example, suppose that the account initially has $X, and then a withdrawal transaction of $Y (where X >=Y) is executed and that the transaction will be compensated later. The interest transaction was applied on a kind of dirty data, and therefore database consistency will not be preserved.

Most of the work assumes rarely changing data; the mobile unit has replica or caching subsystem. And, the mobile replica is logically removed from the master copy of the object and is only accessible by the transaction on the mobile unit, so that they do not consider the case of changing data on the primary server while the transaction processing. In addition, they assume long disconnection or working offline and do not consider short disconnection case.

The Image technique we propose; offers a solution for the preceding problem and gives the opportunity to widely manage the mobile transactions. The Image technique is an optimistic concurrency technique that is used in deferred database recovery and other OS techniques. Image technique uses two copies of data items, the Image copy (original), and the edited copy (current). When a transaction commits, the edited copy becomes the current page, and the show copy is discarded, otherwise, the edited copy is discarded and the image copy is reinstated to become the current page once more.

We assume that the system is partially replicated distributed database system, because it is the most practical environment. We also assume that the mobile unit has a software package that can contact with the primary server and send and receive data from it. We classified the computers that are involved in the update transaction into two groups:

A. The basic group: consists of primary site and mobile unit. They are enough to complete the transaction.

B. The complementary group: consists of all the remaining sites (replicas) that are involved in the update operation and we assume using lazy replication protocol for refreshment.

The technique is basically designed to solve the problem of disconnection in a compound transaction (CT) which consists of two sub transaction groups, one is an independent group that is a collection of independent sub transactions (independent case), and the other is a dependent group that is a collection of vital sub transactions (dependent case). There is no dependency relationship between these two groups, but the compound transaction can include any number of groups.

V. SCENARIO FOR MANAGING DISTRIBUTED TRANSACTIONS USING IMAGE TECHNIQUE

When user enters from one BH to another, its information is stored in image copy & it is updated to original database only when MH logs out or when MH enters in disconnected mode. The relationship among the records in the dataset is independent, so the relationship among the sub-transactions of the program is also independent. Let us illustrate how data is updated in database as user moves from one BH to another.

- Initially Original database contains information of all the users who has logged in till now, & each user has a copy of duplicate database which is initially empty.
- When a user logs in, all the entries corresponding to BH to which it is connected, its login time & logout time are stored in duplicate database.
- Therefore there is no need for MH to connect to BH permanently because every MH has its local database.
- When user logs out or enters in disconnected mode, all the information from duplicate database is transferred to original database & all the contents in duplicate database are cleared.
- This also gives benefit of managing speed mismatch, because information is stored in duplicate database as user switches from one BH to another, but it is updated in single stroke so it results in less communication cost, because MH connects to FH only when updation is required.
- When user wants to read information, it can be read from original database by showing contents corresponding to particular MH.

VI. DISTRIBUTED TRANSACTION PROCESSING MODEL

Distributed Transaction Processing Model is based on overcoming some of limitations of Kangaroo Transaction Model. This Model assumes that there is no need for MH to connect to BH for all the time. We used two databases, one is original & other is duplicate. When MH enters in another BH its information is saved into duplicate database, until MH enters in disconnected mode. When user enters in disconnected mode, its information is saved from duplicate to original database & all information from duplicate database is deleted. Therefore, there is no issue of Service Handoff, because when handoff occurs BH gets its required information directly from database instead of previous BH.

According to Kangaroo model (KM), if a handoff occurs, new BH communicates with the previous BH to get transaction status information. However, in Distributed Transaction Processing model, the new BH will get transaction status information directly from the central database.

A. Assumptions for Implementation

- Only four BHs and one mobile user are used to demonstrate the execution of mobile transaction.
- When the mobile user switches on the mobile unit and connects to the server. The program on the mobile unit sends request to the server to retrieve the current available dataset on it.
- If a disconnection happened through the read set phase, the mobile unit tries to reconnect with the server for a predefined period.
- When the user connects to BH, all updates initially are done in duplicate database (also called image copy) so, there is no need for MH to connect to BH for all the time.
- If the disconnection happened while the user doing his updates on the image copy (because of moving from the current cell to new cell or network failure), the program would not detect the network disconnection, but the operating system of the mobile unit would detect it and gives indication to the user.
- When the user finishes his updates and the network state is active and the connection is available.

We can analyze the movement of the mobile user to 4^{th} areas as shown in Figure 2
Moving from one server area, suppose BH1 to another server area, suppose BH2 i.e. from a->b
Moving from BH2 to the another server, suppose BH3 as b ->c
Moving to another server area as c->d
Moving to area that has no servers at all i.e. in disconnected region as d-> e

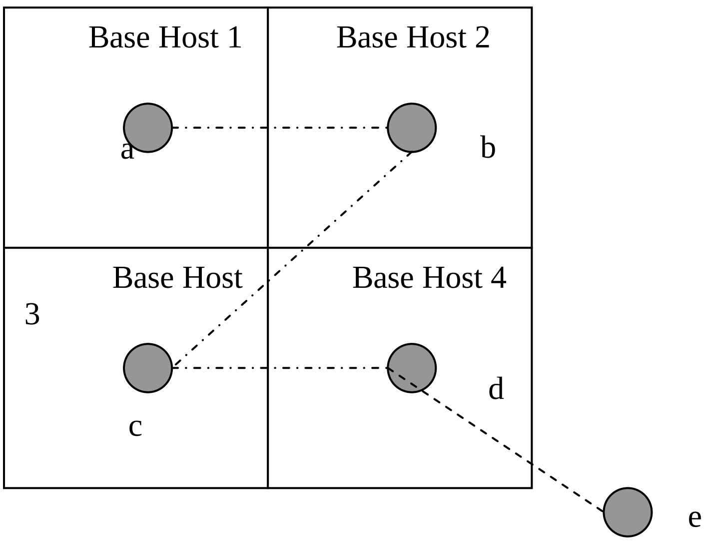
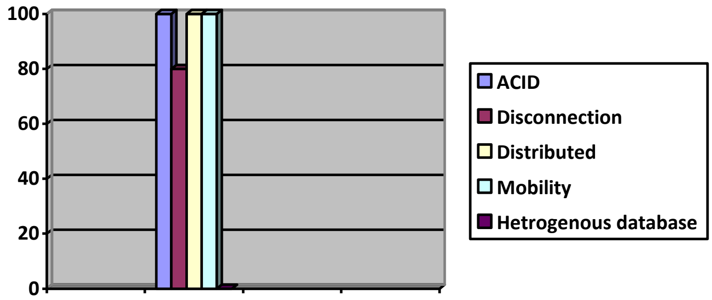


Figure 2 Movement of mobile user

B. Advantages

- *Disconnected Transaction Processing:* This model supports disconnected Transaction processing and information is not lost even when MH enters in disconnected region.

- *Less Communication Cost*: There is connection required b/w MH & BH in this Transaction Model, only when data is to be written to original database from duplicate database, so it results in low communication cost.
- *Freedom from issue of service handoff:* In Distributed Transaction Model information is updated directly in database, so no need of service handoff.
- *Freedom from need of communicating Mobile Host(MH) to Base Host(BH) permanently:* Kangaroo Transaction Model does not deal with disconnected transaction processing, so there is need of communication between MH and BH permanently, but there is no such requirement in Distributed Transaction Model.
- *Differentiates between Short Disconnection and Long Disconnection:* Distributed Transaction Model differentiates between short disconnection & long disconnection. If there is short disconnection, then MH resumes its execution as it reconnects after a short disconnection and in case of long disconnection, user has to login again.
- *Less Efforts Required for data updation:* In Distributed Transaction Model, less effort are required for updation because all information from duplicate database is written into original database at once.
- *Distributed Execution:* Distributed Transaction Model support distributed execution, but sub-transactions are independent in this model, because data is directly saved into database instead of hand over to another BH.
- *To Manage Unpredictable Disconnection:* Distributed Transaction Model is able to deal with unpredictable disconnection, i.e. there is no need to know about disconnection points in advance.
- *ACID Properties are preserved:* This Model ensures all the four properties i.e. atomicity, consistency, isolation and durability.
- *Short Locking At Primary Server:* Lock at primary server is required only when data is to be written from duplicate database to original database in Distributed Transaction Model.
- *Less Load on Primary Server:* In case of Distributed Transaction Model, Mobile Host(MH) connects to primary server (i.e. FH) only when MH logs out or enters in disconnected mode, so there is less load on primary server.



Graph 1 Distributed Transaction Processing Model

VII. CONCLUSIONS

This transaction model is based on concept that supports transaction even in disconnected mode and also ensures ACID properties of distributed transaction system. We have used a technique called Image technique, in which two databases are preserved one is local to MH and other is remote for MH. Initially all updations are done at local database and later on these updations are applied to remote database, so by using this technique, many limitations of Kangaroo Transaction Model and other transaction models have been removed. In this technique we increase the transaction success probability, this by consequence, raises the performance of the system. It doesn't transfer logs or transaction history among sites and it isn't based on compensation concept. It differentiates between short disconnection and long disconnection. It decreases the programming time for applications. So, it is suitable for handling mobile transaction with disconnection. The Image technique used in Distributed Transaction Processing Model can be implemented as stored procedures at the primary server or as a part of the DBMS in many applications.

REFERENCES

[1] P. K. Chrysanthis, "*Transaction Processing in Mobile Computing Environment*", IEEE Workshop on Advances in Parallel and Distributed Systems, pp77-83, 2008.

[2] D. Barbara:"*Mobile Computing and Databases – A Survey*", IEEE Transactions on Knowledge and Data Engineering (TKDE), 11(1), pp 108-117, 2009.

[3] P. K. Reddy and M. Kitsuregawa: *Speculative Lock Management to Increase Concurrency in Mobile Environments*, International Conference on Mobile Data Access (MDA), 2007, pp 82-96.

[4] K. A. Momin and K. Vidyasankar: Flexible Integration of Optimistic and Pessimistic Concurrency Control in Mobile Environments Advances in Databases and Information Systems - Database Systems for Advanced Applications (ADBIS-DASFAA), 2000, pp 346-353.

[5] Lisa Clark, ¨Omer Erdem Demir, "Transaction Management in Mobile Distributed Databases" 2003.

[6] Lars Frank, "*Atomicity Implementation in Mobile Computing*", in Proceeding of 10th International Workshop on Database and Expert System Application , pp: 105, sep 1999.

[7] R. Alonso and H. F. Korth. "*Database System Issues in Nomadic Computing*", in Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 388 – 392, 2003.

[8] M. Satyanarayanan, "*Fundamental Challenges in Mobile Computing*", in Proceedings of fifteenth annual ACM symposium on Principles of Distributed Computing, pp: 1-7, 1996.

[9] Ayse Yasemin Seydim, "An Overview of Transaction Models in Mobile Environments", 1997.

[10] Jim Gray, Andreas Reuter, "*Transaction Processing: Concepts and Techniques*", Morgan Kaufmann Publishers, 1993.

[11] W. Booth, G. G. Colomb and J. M. Williams: "*Transaction Processing Models in Wireless Network*"The Craft of Research, University Of Chicago Press, 2005.

[12] Sharmila John Francis , Elijah Blessing Rajsingh, "*Performance Analysis of Clustering Protocols in Mobile Ad hoc Networks*", Journal of Computer Science 4(3),pp : 192-204, 2003.

[13] E. Pitoura and G. Samaras: "*Data Management for Mobile Computing*", Kluwer Academic Publishers, 1998.

[14] T. Tmiclinski and B. R. Badrinath," *Data management for mobile computing*" Sigaiod Rfcord, 22(1):34-39, March 2001.

