Dual Phased Commit Transaction Model for Mobile Databases

Ayesha Nayyar, Ubaid-UR-Rehman

***Abstract*—Increasing demand of mobility introduced modern trends towards databases. However, issues on transaction management in *mobile* database still persist. This paper analyzes the existing models of transaction management in mobile databases and proposes a new model that deals transactions in two phases; one giving complete rights to mobile host for updating data and second, makes use of the computational capabilities of fourth generation mobile devices for databases manipulation irrespective of other networks issues like connectivity and network interoperability. The proposed model reduces transaction commit time by introducing cache at mobile end and treating disconnected mode similar to connecting mode. It further eliminates the short comings of the existing two tiers, IOT, kangaroo, planned disconnection mode and clustering schemes.**

***Index Terms*—distributed databases; mobile database systems (MDS); global system for Mobile Communication (GSM); dual phased transaction model**

I. Introduction

Data retrieval via PDAs and mobile devices is common practice nowadays. These devices are very useful as they retrieve information using wireless channels from database. But limitations do exist in the flow of information from server to clients. These restrictions also possess limitsthe transaction management. The researchers, practitioners as well as organizations have a shared idea in this regard. The framework of such system includes mobile devices, wired & wireless components along with human resources.

This paper proposes an alteration to the model regardingmanagement of transaction in mobile devices proposed by 'Z. T. Abdul-Mehdi, A. Mamat, H. Ibrahim and M. Deris'. Section II explains the basic mobile database systemarchitecture. Section III summarizes the existing models and in Section IV new model for transaction management in mobile databases is proposed. Section Vdiscusses the comparative studies and finally Section VI concludes the paper.

Manuscript received March 20, 2011.

Ayesha Nayyar is with Department of Computer Science, Military College of Signals,National University of Sciences and TechnologyIslamabad, Pakistan. Phone: +92-300-5330204; e-mail: nayyarayesha@hotmail.com

Ubaid-UR-Rehman is with Department of Electrical Engineering, Military College of Signals,National University of Sciences and TechnologyIslamabad, Pakistan. Phone: +92-345-6067414; e-mail: rehman_ubaid2000@yahoo.com

II. Architecture Of Mobile Databases

Typical MDS not only provides capabilities to access databases via MH. Mobile-user can access database from anywhere anytime, any failure may it be transaction failure, system/media crash occurs, recovery of database is guaranteed by MDS. The architecture of MDS offers facilities like *geographical mobility, connection and disconnection ability, data processing over MH, wireless communication* with server, t*ransparency and scalability*.

A MDS is a client/server distributed or multi-database system based GSMor PCS. The architecture is shown in Fig.1

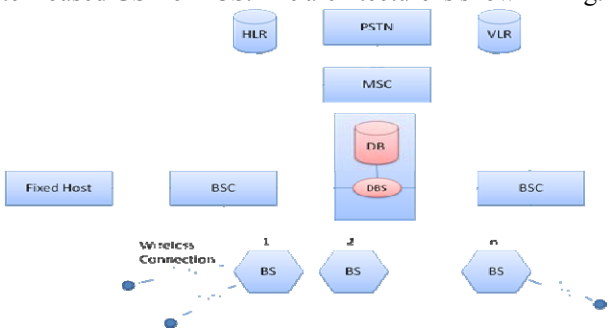

Fig. 1 MDS Architecture

The back bone of the architecture has its roots in traditional PSTN network. Themobile switching center(MSC)is responsible for the connection with PSTN and routing of voice calls, sms and other services. Multiple base station controllers are attached to MSCs. They are responsible for traffic handling, handovers and signaling between mobile phones and networks switching subsystems. The databases are attached to the BSC which is accessed by the MH via BS.

III. Existing Mobile Transaction Models

A. Two Tier Model

Two-tier transaction model [2] isbased on data-replication. This model is also known as Base-Tentative. Database has one master copy and various replicated copies for every data object. There are two types of nodes mobile nodes and base nodes. Two types of transaction exists one Base which operate or work on the master copy and other is Tentative which access or approach replicated data only. The MH can cache data object and tentative transactions updates the replicated data only. These tentative transactions are then converted into base transactions. The base transactions checks for the integrity constraints and updates the respective records over the master data. In case of integrity rules violations, the base transactions are aborted and related tentative transactions are aborted as well.

Also the results of the tentative transactions are visible to the MHs locally but the final update of data is done by base transactions over master data only. In this model two discrete modes of transactions are introduced. Connected and disconnected mode. In disconnected mode if any update is done by the MH it is done by the tentative transaction and is visible to other MHs as well and they will see these respective values. This threatens data base consistency. Also the transaction commit time is large and is executed without the knowledge of any other transaction activities going on in the peer group.

B. Kangroo

Kangaroo [3], a transaction model, gets the progress of MUs also data and apply many of the ACID (Atomicity, Durability Isolation, and Consistency) properties. The parent Kangaroo Transaction (KT) is a set of several sub transactions which are group of reads and writes. These sub transactions are called Joey Transactions (JT) they are local to BS. At the start of KT, BS creates a JT for its execution at MU. IF the MU moves from one cell to another the execution control of this transaction is passed to BS. KT supports the execution of transaction in two modes, split or compensating. If a failure occurs in compensating mode undo all execution of JT also compensate the JTs which were committed previously. Since, the compensating mode is not easy to recognize from the system point of view, so required input information is offered by user for compensating Joey transaction creation. Split mode has been set as default mode for execution. System hardly recognizes the mode of compensation. In order to compensate Joey transaction user has to provide valuable input. Split mode is default execution mode. Failure of JT is the failure of default mode. In this case there is no recreation of JT and previously committed JT are not compensated. JTs can be processed in serial manner in compensating mode but not in split mode.

C. Isolation only Transaction (IOT)

Isolation Only Transaction (IOT) [4] detects read and writes conflicts. These conflicts are noticed and recorded as a sequence of transactions. At reconnection transactions are compared with server state through global verifiability criterion. Conflicts can be detected by comparison of server to the modified file. IOT is an extension of UNIX and CODA which considers write/write conflicts only. On a successful validation transaction is committed at server. In other case the IOT automatically tires to resolve the conflict. Correct results are reconstructed by the resolution of the recorded changes serially.

For example a user raised a transaction T and its execution has to be performed on client. Remote files are retrieved from client's disk cache. Partial execution results are not viewed on server. After the execution of T, state changes to waiting or committed state depending on the condition of the connectivity. T is committed for complete access of the file only. Client maintains server connection for each file accessed by T. Results is viewed on servers after T has been committed. T goes to waiting state if it was not committed and then validated later. Client holds results temporarily in its cache.Temporary results are visible to subsequent process on the same client only. T is validated accordingly to the consistency criterion. T is integrated and committed to the server otherwise it goes to resolution state. After the resolution of T, manual or automatic, result is committed to the server. In some cases for example, rejection of transaction, model adopts specific applications to resolve the disputes. This model suggests the utilization of queue to process transactions.

The system is transparent. It automatically detects and resolves the conflicts. But in some cases system gets defaulted and has to be repaired manually.

D. Planned Disconnection Mode

Planned disconnection mode [5] presents four diverse disconnection models for mobile databases. These include optimistic checkout, relaxed checkout, checkout and basic sign-off. This model supports planned disconnection database operations for one MH only. Mobile data sharing protocol can deal with, single or multiple hosts as well as planned and unplanned disconnections.

Mechanisms like Two Phase Lock (2PL) for concurrency control are employed if transaction processing is to be executed to every mobile host locally. Pre-commit record is transmitted to other MH using message exchange like epidemic. All MHs then synchronize through epidemic. During connection transactions can be processed by multiple MHs. There are situation when one MH requests disconnect while rest MHs continue transactions. Sign-off disconnection mode is used in such situation. It uses the benefits of planned disconnection.If a MH request disconnect it permits another MH to perform functions as its replacement. The substitute MH performs transaction with other MHs on the behalf of disconnected MH. The disconnected MH can carry out read only transactions. On reconnection MH will assume control of all processing from its substitute MH and takes updates from it. In check out disconnection mode if a MH wants to disconnect or it has to update an object, it locks the related items. These objects can only be checked out by one site. Access to these objects, in database, by other MHs is restricted. Relaxed check out disconnection mode is used where the access to the locked items is required by other MHs. In Optimistic checkout mode, both connected and disconnected MHs can access the whole database.

The dilemma of this model is that only one MH is permitted to have copy of a data object.

E. Clustering

In Clustering Model [6] a completely distributed system is assumed. The transaction model was designed so as to maintain the database consistency. In this the database is separated into the clusters and mobile transaction is decomposed to a set of strict and weak transactions. This breakdown or decomposition is made on the basis of requirementof consistency. Moreover the write and read operations alsoare grouped as strict operations or weak operations. The strict has a database wide access while the weak ones are only allowed the data element access that belongs to the same cluster. That means two copies are maintained one weak and other strict.

The model makes transaction processing possible on MH

while in disconnection mode. It replicates the data on the MH locallythrough clusters to provide the ability to direct or operate data during disconnection. As we know cluster groups together the related data and so a MH can be seen as cluster. When disconnected operations have been performed on the cached data (locally), the MH must connect again to reconcile its modifications or changes.MH performs usually like a device that always connected. Two types of transactions are initiated; one is a Weak transaction that writes and reads on local copy while in disconnected mode that is local commit. These are committed in related clusters only and remaining clusters accept these later than reconciliation that is global commit. Other type is of strict transaction write and read data on every cluster during connection mode. The need of reconciliation of week transaction results within the global cluster is to avoid conflict with strict transaction results. The implementation details are not mentioned in the paper.

To preserve the atomicity property in each cluster the transactions that are weak are globally committed only if there is no conflict with the updates of transactions that are strict. So, weak transactions are helpful when while handling data seldom accessed. Also isolation is guaranteed using locking protocol initiated for weak transactions. The compatibility of lock among weak and strict operations is ensured by conflict tables that depend on the application requirements semantics.

The problem with this model is that whenever the network is divided weak transaction can progress or precede causing data inconsistency and so the transaction is abort.

F. Model for Transaction Management in Mobile Databases

The model proposed by Ziyad T. et al [7] is shown in Fig 2.Themain idea behind scheme is to execute the transactions at BS as well as on MH. Number of MH can be attached to the single BS via wireless link. The BS is further attached to the corporate database server. Whenever the transaction is initiated at the MH two types of transactions are introduced, pre-committed transactions and request transactions. The former transactions that can be committed wholly at the MH while later transactions cannot be entertained by the MH are sent directly to the BS where they are executed and committed.

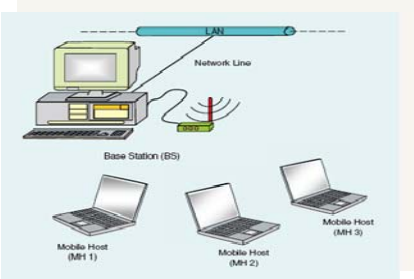

Fig.2 Transaction Model

The pre-committed transactions are then re-executed as base transactions (BT) and the results are written over the corporate databases.Serialization of BS is done over themasterdata's copy, ensuring data consistency. Each BS provides MHs with the set if variables (∂, t). This function refers to the maximum number of attempts for data access that a MH is allowed in given time periodt. If the count increases more than ∂, then the transactions are not pre-committed at MH but are sent as request transactions at BS. These BTs are executed there and results are sent to the MH.

∂ is calculated using equation 1.

$$\partial = [(AV + (€ * r)) * d_i/(m_i + n_i)] \quad (1)$$

The variable *r* represents the request of the data item value by MH, initially *r=0 and* increases by one whenever there is a new request. Maximum value of r=9. AV=0.5, €=0.05, d_i be the current data value D_i, n_i is the number of MHs requesting the data item, $m_i$is the number of previously connected MHs.

IV. DUAL PHASED COMMIT TRANSACTION MODEL

The dual phased commit transaction model for mobile databaseis an advancement of the transaction model proposed by Z. T. Abdul-Mehdi et al. [7]

The system architecture is shown in Fig. 3. As the figure depicts the system model consists of a base station (BS), corporate databases and mobile hosts (MH).The MHs are connected to the BS wirelessly. Both the MH and BS have the permissions to update data. When the MH updates the data, it is pre-committed at its end and then is executed at the BS as base transaction (BT). The results of the pre-committed transactions are available to the MH and it can further work over it. The BTs are serialized on the master copy of data, ensuring data consistency. BS has the responsibility to hold and manage data. The data items that can be updated by MHs are distributed among them. The MHs are given time based permissions to connect, disconnect and reconnect to the BS respectively.This increases network efficiency and also restricts bottleneck by the MH having large amount of data at the BS.

Since the model is designed for fourth generation mobile devices that have reasonable computational power and memory space. The usage of buffer at the MH for executing transactions is proposed. The connected MHs are grouped andtheir randomdisconnection from the network increases the network efficiency and thus in turn effects the transaction commits time positively.

Fig.3System Model

A. Transaction Execution Mechanism

1) Phase1 Connected Mode

When the MHs are connected to the BS, the BS adds an entry for every MH that connects with it.Also MH assigns certain memory in kilobytes of space as buffer space in its cache where it can store data related to the active transaction. Let D be the set of data item and d_i is the current value of the data D_i.n_i is the number of mobile hosts requesting the data

item and m_i is the previously connected MH. A portion of d_i called ∂i is assigned to each MH and the balance is kept at BS. ∂_i defines the upper bound of the update or change of a data item D_i. The value of ∂_i is expressed as $0<\partial_i<d_i$. If the transaction changes the data item at most ∂_i and is in bounds of function (∂,t), then the MH does not need to wait for the results from BS and its free to pre-commit. But if the limit of (∂, t) is exceeded, the transaction is sent to the BS and is not executed at MH. Such transaction is called request transaction (RT). The BS issues the functionP = {∂, t, T_d, T_{rc}} to each MH. Value of ∂ for each data item D_i is calculated using function F (d_i, n_i, m_i). So,

$$\partial_i = f_i(d_i, m_i, n_i)$$
$$= [(AV + (€*r))*d_i/(m_i+n_i)]$$

Variable represents $(r+1)^{th}$ request over data item. Initially r=0, value of r increases by onewith every new request.Limits of r can be expressed as AV and € is the balancing factor. Value of AV is equal to 0.5 and that of € is 0.05. The explanation of this balancing factor is beyond the scope of this research. t is the time during which ∂_i is valid for a particular MH.t limits can be expressed as 0 and t T_{rc}.T_{dc} is the time after which the MH will get disconnected from the BS automatically,0 T_{dc} .

'T_{rc}' is the time duration after which the MH will reconnect to the BS after disconnection, T_{dc} T_{rc}. Upon reaching T_{dc}, if the MH still has the transaction in progress, T_{rc} is assigned and thus at T_{rc}, the MH reconnects to complete its transaction. T_{rc} is bound by the upper limit of t. Note that the BS will not update the value of data until the time limit t has elapsed.

Phase: 2 Disconnected Mode

If the MH is in disconnection state, the pre-committed and request transaction are treated in the same way. The presence of buffer offers data storage for pre-committed transactions. The mobile host makes copy of the data item D_i over which it is performing transaction. Thus even in disconnected mode too all the transactions are pre-committed at MH and it does not have to wait for the results from BS.The states of such transactions are maintained and thus on reconnection these pre-committed transactions are committed. It must be kept in mind that the timings for disconnection and reconnection are bound by (t) limit. This will ensure that within the valid time all transactions are executed. BSs are scheduled to issue new (∂, t) value after each MH time limit is expired.

The transactions at database are serialized and advance queuing techniques are used to maintain data consistency.

2) Transaction Execution at Base Station

The BS will perform the main tasks

- ∂ Calculation for each data object.
- Associating $\partial$with *t*.
- Distributing (∂, t) for every data item $D_i$to each MH.
- Transaction execution in aserial manneron master data item to maintain data consistency.
- Associating T_d (disconnect time) and T_{rc} (Reconnect time) values to the set of (∂, t).
- Committing the pre-committed transactions.

3) Transaction Execution At MH

The MH performs following tasks

- Executing transactions.
- Saving transaction in disconnection mode for forwarding to BS at reconnection.
- Calculating valid values of ∂.

For this purpose a variable X is maintained, which expressed the accumulated number of successful commit. This value is calculated as X=X+ (∂-t_1). If X > ∂i. then t_i transaction is stopped or blocked at MH and as request transaction submitted to the BS. Initially X=0 and increases by ∂-t_i (where t_i is any transaction).

V. Discussion and Experimental Results

The proposed model in an extension of the model proposed by Ziyad T. et al [7]**.** In his model if count increments than **∂,** the pre-commit transactions are treated as request transactions at BS and results are sent to the MH. This makes an overhead which is removed by our model. In our model the pre-commit transactions on disconnection is not treated as request transactions. These pre-committed transactions are sent to BS on re-connection where they are executed as BT. The results are then committed to the corporate databases. This is possible by storing data of interest in the buffer space allocated at the MH for the particular transaction. This feature also provides the advantages of group transactions. The model decreases transaction commit time positively and enhances network throughput.

Fig. 4 shows the comparison of model proposed by Ziyad T. et al [7] and Dual phased commit transaction model for mobile databases in connected mode and Fig. 5 shows their comparison when the MHs are re-connected after disconnection.

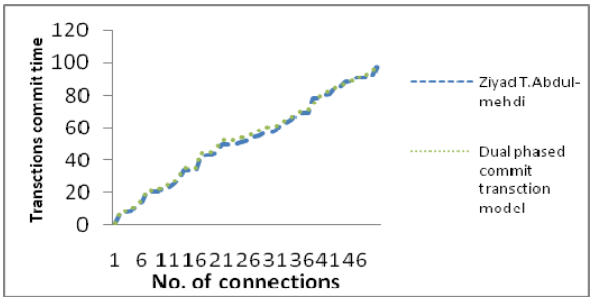

Fig.4 Connected Mode

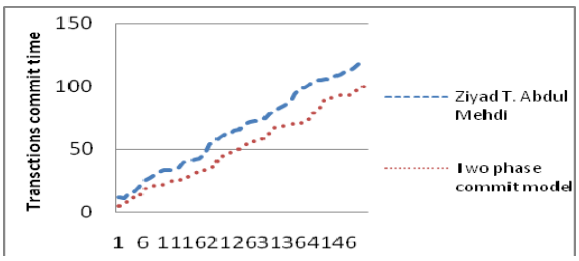

Fig. 5 Reconnect Mode

When the mobile is in disconnected phase the data processing activity is almost negligible at MH in by Ziyad T. et al's [7] model, but since we are dealing with the fourth generation mobile devices, they have the ability of computation and thus the transactions continue to progress at the MHs that are further processed upon re-connection at BS. This lessens the differences between connected and disconnected mode. Though the mode alterations are useful for creating group transaction, reducing such difference affects the data processing activities. In our model it increases with time and thus the resource usage of MH is also

increased. The comparative graph with that of Ziyad's model is shown in Fig. 6.

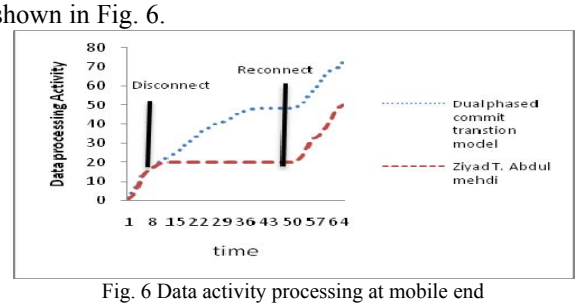

Fig. 6 Data activity processing at mobile end

Some sort of inconsistency is introduced by Clustering model [6] as database state can be updated by only strong transaction. Thus weak transactions are aborted causing inconsistencies. In the proposed model both pre-committed and request transactions immediately updates data. This reduces transaction commit time and helps in maintaining consistency.

The shortcomings in acidity, durability, consistency and serializability are removed by permitting BS only to update the master copy of databases**.** Introduction of queuing for transaction serialization also solved the problem in IOT [4].

In planned disconnection model [5] there is a problem of data loss as only one MH can have the data object copy. In our model several MHs can have the same copy of data and only the BS can update the data only and has permissions of update.

Though the model presented by Ziyad T. et albids ACID properties, but the transaction commit time is a bit longer. Overhead of re-executing the request transaction is present. "Dual phased commit transaction model for mobile Database"**,** is and advancement of Ziyad T. et al 'model. Our model offers memory usage of MH in disconnected mode which helps in reducing network load and transaction commit time.

The table showing comparative analysis of acid properties of the existing models is shown in Table 1.

VI. Conclusion

The paper renders an insight into various transaction management models for mobile databases. After analyzing the existing models a new model is proposed which tends to reduce the flaws present in the studied models. Since the model is tested in lab environment it's in- field implementation is left for the researcher, this model is an open area for researchers to critique over the issue. In the proposed model the phases of transactions are well discriminated and proposes maximum facility for MHs to process transactions. The transactions speed is raised by introducing group transaction and time based connection and disconnection of MH with the BS. Some of the experimental results are presented and more analysis is in progress to validate the correctness of the proposed model further.

TABLE I. Analysis Of Transaction Properties Of Existing Models

Model Name	ACID PROPERTIES OF DATABASE			
	ATOMICITY	***CONSISTENCY***	***ISOLATION***	***DURABILITY***
Two tier model[2]	Y	N	Y	N
Kangaroo[3]	N	N	Y	N
Clustering[6]	Y	N	Y	Y
Isolation only transaction [4]	N	N	Y	Y
Planned disconnection mode[5]	Y	N	Y	Y
Mobile transaction model by *Ziyad T. et al*[7]	Y	Y	Y	Y
Two phase transaction model (proposed model)	Y	Y	Y	Y

References

[1] Vijay Kumar, "Mobile Database System," Computer Science and Informatics, University of Missouri-Kansas City, John Wiley and Sons, pp. 125-129, 2006.

[2] J. Gray, P. Helland, P. O'Neil, and D. Shasha, "The Dangers Of Replication And A Solution," ACM SIGMOD Conf., Montreal, Canada, pp 173-182, 1996.

[3] M. H. Dunham, A. Helal and S. Baqlakrishnan, "A Mobile Transaction Model that Captures Both the Data and the Movement Behavior," ACM/Balter Journal on special topics in mobile networks and applications, vol. 2, no. 2, 1997.

[4] R. Kong, S. Lau, and D. New, "Mobile Data Access: Disconnected And Partially Connected Approaches," Dept. of Computer Science, Univ. of California, San Diego, La Jolla, CA, 2001. DOI= http://cseweb.ucsd.edu/classes/wi01/cse221/OSSurveyW01/papers/kong,lau,new.mobile_data_access.pdf

[5] J. Holliday, D. Agrawal and A. El Abbadi, "Disconnection Modes For Mobile Databases," J. Wireless Netw., vol. 8, no. 4, pp. 391–402, 2002.

[6] Pitoura and B. Bhargava, "Data Consistency In Intermittently Connected Distributed Systems," IEEE Trans. Knowledge Data Eng., vol. 11, no. 6, pp. 896–915, June 1998.

[7] Z. T. Abdul-Mehdi, A. Mamat, H. Ibrahim and M. Deris, "A Model For Transaction Management In Mobile Databases," IEEE, vol. 29, no. 3, pp. 32-39,May 2010.

[8] M. Bowman, S. K. Debray, and L. L. Peterson, "Reasoning About Naming Systems," ACM Trans. Program. Lang. Syst. vol. 15, no.5, pp. 795-825, 1993.

[9] Ayesha Nayyar,Ubaid-ur-Rehman, "An Advance Two Phase Commit Transaction Model For Mobile Databases" , preceedings of ICCAE 2011.