ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

Adaptive M-Shadow Model for handling Mobile Database Transaction Processing

Romani Farid Ibrahim

Department of Computer information systems, Faculty of Information Technology,

Al Hussein Bin Talal University, Jordan

Abstract: Recent advances in wireless communications and mobile devices such as laptops, mobile phones, PDAs, etc, and their cheap price and the emergence of many mobile applications have provided users with the ability to access data from anywhere. In this paper, our concern is the management of disconnection in mobile transaction that access temporal database and/or spatial database. M-Shadow (Mobile-Shadow) handles a compound (or a group) transaction that consists of groups of subtransactions which may be independent or dependent. We adopted the M-Shadow to handle partially dependent subtransactions and to be implemented as one application handles the three cases together (independent and/or dependent and/or partially dependent), and taking in consideration the location dependency. M-Shadow uses of a notation of actionability, which differentiates the actions to be taken during the transaction's validation phase according to the types of affected attributes. We extended the actionability data types to include Change-Passing and Location-time attributes. The M-Shadow technique increase the success probability of transactions processed under optimistic concurrency techniques.

Keywords: Concurrency Control, Mobile Database, Transaction, Shadow paging, Saga, Caching, Compensation, Vital and Non-vital transactions, Temporal database, Spatial database, GPS.

I. INTRODUCTION

The mobile computing and moving objects area is very interesting and active area of research, because it includes many other subjects as networking concepts, operating system concepts, database concepts, etc. Accessing data anywhere-anytime-anyway will be real but this must be consistent. The mobile database, or embedded database on a mobile device, is starting to become an important player in all practical fields, for example, business, traveling, police, military, medical, etc. The data will be entered approximately in its real time, no delay between the events time and the entering time to the database.

Mobile transaction is a transaction performed with at least one mobile host takes part in its execution [6]; 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 [7]. 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.

As an example of applications that uses mobile transaction, we are considering mobile hosts are laptop computers belonging to members of a big salespersons team. The salesperson performs a compound transaction that handles a customer big order which consists of a group of independent sub-orders and/or a group of dependent sub-orders and/or a group of partially dependent sub-orders. Also these transactions can be location dependent or location independent.

We view a transaction as a program in execution in which each write-set satisfies the ACID properties [1], and the program that updates the database as a three folds module (phases): reading phase, editing phase, and validation and write phase. The main question we attempt to answer in this paper is, if the data on the primary server has been changed while the mobile unit (MU) is disconnected or working offline, how can the transaction continue its work?

The proposed M-Shadow technique is an optimistic concurrency technique constructed on the shadow paging technique that is used in deferred database recovery and other OS techniques. Shadow paging technique uses two copies of data items, the shadow 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 shadow copy is reinstated to become the current page once more.

This paper is organized as follow: Section I gives the introduction of the mobile database system and mobile transactions. Section II is helpful to understand the background of related work. Section III explains the important points we considered to propose the new model. Section IV explains the M-Shadow technique. Section V explains summary of the implementation and performance of the proposed technique and the last section VI concludes the paper and followed by the references.

II. RELATED WORK

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

Most of the work handling mobile transactions as (Kangaroo, Reporting and Co, Moflex, Escrow techniques…) assume that the handoff process is under the mobile support station (MSS) responsibility [9], and the mobile support stations has the capability to transfer control and transaction history among servers while handoff procedure as [5], [7], [8]. 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 [9]. 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 [10]: 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. If another transaction commits applying an interest rate on the balance before the compensation has been performed (i.e. when the account has $(X-Y). The interest transaction was applied on a kind of dirty data, and therefore database consistency will not be preserved.

Most of the papers assume rarely changing data (Insurance data, Patients data, etc); 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 [11], 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.

III. IMPORTANT CONSIDERATIONS

In optimistic methods using shadows, transactions are dependent on all data items by the same degree. A minor change in an item is sufficient to abort a transaction handling hundreds of actions on thousands of data items. Consequently, the probability of a transaction to fail is very high. This failure probability increases with the increase of the number of data items, the disconnection time, and the number of concurrent transactions. This is why the shadow technique is not frequently used in transactions management.

The M-Shadow technique we propose; offers a solution for the preceding problem and gives the opportunity to widely manage transactions of difficult types such as long and/or mobile transactions. In M-Shadow technique, transaction's validation is not tightly coupled to the eventuality of encountering modifications (done by other transactions) on the values of one or more of its data items.

In this section, we describe the important points we considered to design our technique for handling mobile transaction with disconnection. Which are: the enterprise constraints acceptance range, and the effects of attributes types on the transaction behavior (actionability), linear and non-linear applications, and the structure of M-Shadow transaction.

A. Enterprise Constraints Acceptance Range

Enterprise constraints also known as business rules, which are additional rules specified by users or database administrators that the data must satisfy [14]. Usually, enterprise constraints include relational operators as (<>, <,>, ≤, ≥), which has a range of values that the data-item can be changed within it. For example, a sold-amount value can be assigned any value from a range of values.

By using this property of enterprise constraints, in addition to, the characteristics of the attributes, we can build an algorithm that allows transactions to continue their works even if the shared data items at the primary server have been changed. So that we avoid roll-backing of transactions, if the changes are within the acceptable range of the data-item.

B. Actionability and Transactions Behavior

In M-Shadow technique, transaction's validation is not tightly coupled to the eventuality of encountering modifications (done by other transactions) on the values of one or more of its data items. Transaction behavior at run time depends on some characteristics of its set of data items. We use a new notion called actionability to describe how a transaction behaves if a value-change is occurred on one or more of its attributes during its processing time and by other transactions. Other than Key attributes *(K)*, actionability classifies the data items used by a transaction into five types: change-accept, change-aware, change-reject, change-passing *(P)* and location-time *(L)* attributes.

Change-Accept *(A)*: Any attribute retrieved during the read phase to complete and explain the meaning of the transaction. If it is potentially changed (by another transaction) while the transaction is processing, it does not have any effect on the transaction behavior.

Change-Reject *(R)*: This type of attributes is subject of periodical changes (e.g., Currency values, Tax rates, etc.). The value of such attribute remains constant for long period. But once it is changed during the transaction life time (by another transaction), it affects severely the transaction behavior.

Change-Aware *(W)***:** This type of attributes is subject to change more frequently by different concurrent transactions. A modification on the value of this type of attributes may be accepted if the new value still in the acceptance range. Otherwise, the transaction aborts.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

Change-Passing *(P)*: this type of attributes is not basically part of the transaction data, but the result of the transaction processing is passed to this type of attributes. For example, in an insurance company (or many other applications) all different departments are related through the financial department, so that, all insurance transactions in all departments should pass their financial values to the financial attributes. Usually this subtransaction is succeeded because it only increases the financial attributes by the new amounts and the previous change and the current values of this type of attributes doesn't effect on the transaction data or behavior. But if the subtransaction that changes their values is failed for any reason, it causes the main transaction to fail.

TABLE 1: ACTIONABILITY TRUTH TABLE

Change in				Integrity Constrains Violation	T Succeed
Change-Accept Attribute	Change-Reject Attribute	Change-Aware Attribute	Change-passing Attribute		
Y / N	N	N	Y / N	NA*	Y
Y / N	Y	N	Y / N	NA	N
Y / N	N	Y	Y / N	N	Y
			Y / N	Y	N

***NA means Not Available**

The previous three types of attribute actionability (Change-Accept *(A)*, Change-Reject *(R)*, and Change-Aware *(W)*) are to be declared for each transaction type. If omitted, the complete set of attributes will be handled as Change-Reject type (the default actionability type), a case in which the M-Shadow works like the traditional Shadow technique. Also, they are retrieved at the read phase to be edited and to apply the function of the transaction on it. It is also important to note that a transaction may generate a new data item *(G)* as a function of the three previous types of attributes. The M-Shadow technique handles these attributes exactly as before:

- If a Change-Reject attribute(s) is modified during the transaction processing, the complete transaction aborts.
- But else, if a Change-Aware attribute(s) is the modified attribute and the changes are within the acceptance ranges, the transaction is recalculated and continues, otherwise it aborts.
- But else, if a Change-Accept attribute(s) is the modified attribute, the transaction continues and writes values.

Table (1) illustrates the applied validation rules. If the Change-Accept attribute and the Change-passing attributes are changed or not, it doesn't have any effect on the transaction behavior that updates the Change-Aware attributes. Also, Change-Accept attributes are very rarely changing attributes, for example, item-description, employee-name; Birth-Date, etc., are approximately fixed value attributes.

Rule: If T1, T2 are concurrent transactions, T1 changes a shared Change-Reject attribute and T2 changes a shared Change-Aware attribute that belong to a normalized database then:

- If T1 commits before T2 then T2 must abort.
- If T2 commits before T1 then T1 can continue its processing.

The reasons behind using the actionability include:

- A transaction usually update a part of the data set it uses, the other part of the data elements is asked by the transaction to control the transaction. These data items are read only items and a change in such elements should not prevent the execution of the transaction.
- Our concern is on the transactions that update Change-Aware attributes, which have acceptable range. An encountered change in these attributes may affect the outcomes of the transaction but not aborts entirely its execution.
- The usage of mobile transactions is still limited to salesperson and inventory applications which are, by nature, applying short transactions with little attributes. This fortunately complies well with the M-Shadow concept.

C. Location dependent transaction

To handle location dependent transactions, we assume that the mobile database system of the company can connect to a Global Positioning System (GPS) that determines the location of the mobile unit (latitude/longitude) and answers the location dependent queries [16]. For example, if the salesperson decided to visit his customers according to their nearest from his current location and typed the query: Find the nearest customer (address) to my current location? The problem here is that the customer address is in the temporal database not in the spatial database. To answer this query, the query analyzer should perform the following steps: retrieve the customers primary keys and addresses (assume streets), pass it to the GPS, the GPS compares these data with its database and arranges the data according to the current location of the mobile unit and returns the result to the mobile database system. Based on the returned result from the GPS, the customer details are retrieved from the mobile database system and are passed to the mobile unit.

We assume that the table that includes the history of transactions should include attributes to store the X, Y (latitude/longitude) location of the mobile unit and the time of issuing the transactions. These data are stored for the purpose of retrieving and for future analysis. These data does not have any effect on the transaction behavior or on other attributes that are accessed by the transaction. We call

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

these data a transaction geographic data and the other data a transaction basic data to differentiate between them. These data are different from the data that describe location and time attributes related to a specific transaction as: location of shipment, location of receiving, airplane departure time, etc. So that, we extend the actionability classification of attributes to include this type of attributes by adding a new type of attributes called location-time-attribute (L).

D. *Actionability advantages*

The classification of attributes according to the actionability type has the following advantages: The DBMS can perform the update process automatically based on the actionability type of the data for all applications. A reasonable increase in the succeeded transactions ratio. This technique can be useful for some types of real time applications.

E. *Linear and Non-Linear Applications*

Most of papers that handle transaction processing assumed implicitly that all the applications increase or decrease the attributes (in our notation change aware attributes *(W)*) by Δ value. But the applications that update a change aware attributes *(w)* can be classified based on the mathematical functions that is used to calculate the new values of the change aware attributes into two types:

- Linear transactions: that use the linear function $f(x) = mx + b$ for the calculation of the new value of change aware attribute. We assume that the m value usually equals 1, and the b is the value of the changes that transaction are performed on the w attribute, which is known as delta $\Delta(w)$, and the function can be written as $f(w) = w \pm \Delta(w)$, where w is the original value of the change aware attribute. The function includes only add or/and subtract operations *(+, -)*.
- Non-linear transactions: that use other functions that differ from the form of $f(w) = w \pm \Delta(w)$. They can include the functions: Power, Div, Multiply, Len, Log, Sqrt, Sin, Cos, Tan, etc. The semantic of these types of applications require to recalculate *f(w)* according to the current *(w)* at the validation and write phase at the primary server.

IV. THE ADAPTIVE M-SHADOW MODEL

In this section we explain the structure of the M-Shadow transaction, the processing of the validation test, summarizing the M-Shadow technique steps for linear transactions, how the system determines type of transactions (linear or non-linear), and advantages and limitations of the M-Shadow technique.

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:

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

A. *M-Shadow Transaction Structure*

M-Shadow relaxes the original saga constrain which is either all subtransactions completed or all subtransactions compensated for their effects on the database. By collecting subtransactions into groups and handles each group according to the semantic of the relationship among them without using compensation, therefore a subtransaction effects is limited to its group not to the entire M-Shadow transaction.

- If the subtransaction alone is independent, then, when it is grouped with other subtransactions in one compound transaction (CT), it has three cases:
  - It does not lose its independency property, so it can commit alone.
  - It loses its independency property, and it has a dependency relationship with its CT. IF it fails, the CT fails, if the CT fails for any reason, the subtransaction fails also.
  - If it is a non-vital subtransaction, it can abort alone and doesnot effect on vital subtransactions of the CT and the CT can commit without it.
- No use of the compensating subtransactions

Figure (1) shows an example of M-Shadow transaction which consists of three subtransactions groups. The first group is a group of logically independent subtransactions (S1, S2, and S3), the second group is a group of logically full dependent subtransactions (S4, S5, S6) and the third group is a group of partially dependent subtransactions since s8 is a non-vital transaction while s7 and s9 are vital subtransactions. There is no dependency relationship among these three groups, but the compound transaction can include any number of groups.

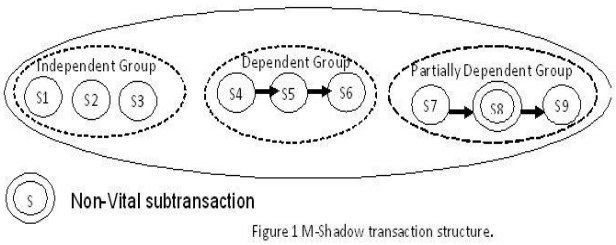

Figure 1 M-Shadow transaction structure.

B. *Description of Validation Test*

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

The validation test compares the original values of some data-items with its current values on the primary server, which succeeds in three cases:

- No change, which means that the original values are equal to the current values on the primary server.
- Constrained change, which means that some Change-Aware attributes has been changed by other transactions during disconnection (working offline) time but still these changes within the integrity constraint acceptance range.
- Insignificant change, which means that some Change-Accept attributes has been changed by other transactions during disconnection (working offline) time or during the execution of the transaction, but these Change-Accept data-items does not effect on the current transaction.

The validation test fails in the following two cases:

- Significant change, in which we detect that some Change-Reject data items have been changed during the transaction processing and/or disconnections.
- Out-of-Constraints change, in which we detect that one or more Change-Aware data items have been updated in such a way that the global changes put the stored values out of the acceptance ranges.

C. *Summary of the M-Shadow Technique Steps for Linear Transactions*

In Location Dependent transaction

- Get mobile unit location data (x_loc, y_loc)
- Get the server identification data
- Retrieve datasets from the current server of the current cell (called the home-server of the transaction or transaction primary server).

In location independent transaction

- Retrieve the current dataset from the primary server (Reading phase)

In both cases of location dependency

At Mobile Unit side:

1. Copy the retrieved dataset as a shadow copy.
2. The user edits the dataset on the shadow copy [modify, add, delete] (Editing phase)
3. Send the original read-set, the edited-set (shadow copy changes), the read-query and, and the update query to the primary/home server (subtransaction by subtransaction).

At Primary Server Side:

4. Implement the validation and write phase:
   - Call validation-write-1 procedure (as a part of the DBMS) or
   - Call validation-write-2 procedure (as a stored procedure at the primary server).

1) *Independent Case*

5. If one subtransaction fails (disconnection, integrity constraints, etc.)

At Primary Server Side:

- Discard the current write-set subtransaction.

At Mobile Unit side:

- Removes the subtransaction shadow data-set from the shadow copy.
- Send next subtransaction data to the primary server.
- Short disconnection (the user doesn't close the program which means all variables and data-sets still available in the main memory): Try to reconnect.
- Long disconnection(the user wants to close the program): The program saves the data-sets (the original data-set and the remaining elements of the shadow data-set) as XML files on the mobile unit secondary storage to be retrieved at the reconnection time.

When reconnection with the primary server is available:

After short disconnection:

The program resends the write-set data for the subtransaction, which the disconnection happened through its update only. The primary server restarts the write-set subtransaction as in step 4.

After long disconnection:

The program loads the XML files and starts a new independent write-set group transaction for the loaded data-sets (original and shadow) as in step 3.

2) *Fully Dependent Case*

6. If one subtransaction fails:

At Primary Server Side:

- Rollback the current and all the previous write-set subtransactions of the group.

At Mobile Unit side: because of

- Integrity constraints violation: Drops its data-sets and clears the memory to start a new transaction.
- Short disconnection: Try to reconnect.
- Long disconnection: The program saves the data-sets (the original data-set and the shadow data-set) as XML files on the mobile unit secondary storage.

When reconnection with the primary server is available

After short disconnection:

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

The program reissues the dependent-write-set group transaction as a new transaction as in step 4.

After long disconnection:

The program loads the XML files and starts a new fully dependent write-set group transaction for the loaded data-sets (shadow and original) as in step 3.

3) Partially Dependent Case

The transaction processing for partial dependent group is similar to the full dependent group procedure with neglecting the non-vital subtransactions in case of its failure (ex, S8). The partially dependent group transaction fails in case of failure of any of their vital transactions (ex, S7, S9).

4) Location Dependent Case

- In general, as the validation and write phase of the M-Shadow model for location independent data.
- In addition, in long disconnection case, the mobile unit saves the transaction home-server identification data as XML file.

At reconnection time

- After long disconnection case, the mobile unit loads the datasets and the transaction home-server identification data from the XML files.
- Connect with the home-server of the transaction from the current cell.

Validation-Write Procedure-1 (A General Validation Algorithm to Be Put as a Part of the DBMS)

Validation-Write-Phase (Record original, Record shadow, String read-query, String update-query)

In what follows we show the core functions of the technique, which use the actionability rules to perform the validation test. Its inputs are original data-set, shadow dataset (shadow-rec), read-query, update query, and the actionability types for attributes if they are not declared while tables creation. If the validation test succeeds, the transaction commits, otherwise the transaction aborts.

```
Aware-Update (integer flag)
{
For each change-reject-attribute(i) in shadow-rec
  If  Current.R(i) <>  Shadow.R(i)    then
     Flag = -1
     Goto par-out
  End if
Next-For
 For each change-aware-attribute(i)   in shadow-rec
    ΔW(i)   = Shadow.W(i)   - Original.W (i)
    Current.W(i)   = ΔW(i)   +  Current.W(i)
    If (check-constraints(current.W(i)   ) = False ) then
        Flag = -2
        Goto par-out
  Next-For
Par-out:
Return (flag)   }
```

Validation-Write Procedure-2 (Stored Procedure at the primary server)

```
Sub Validation-Write (ti)
 {
Begin write-set subtransaction (ti)
   Hold exclusive lock (ti)
     Read data from active database for (ti) as current
     If  change-reject data-item is changed then
        Rollback transaction (ti)
     Else
        Calculate Δ(x)   = Shadow(x)  - Original(x)
        Current (x)   = Current (x) +  Δ(x)
        Check-validity (Current (x))

        If check-validity success then
          Write shadow data-set to the current database
            Commit Trans (ti)
         Else
              Rollback transaction (ti)
        End IF
     End IF      }
```

Table 2 shows an example to describe how the validation and write phase can be applied and assume linear transactions for simplicity. The example shows a bank transaction that transfers $400 from account X to account Y. We use the notations of actionability, K denotes the Key attribute, A denotes a Change-Accept attribute, R denotes a Change-Reject attribute, W denotes a Change-Aware attribute, G denotes a generated attribute, and the subindexes o denotes the original value, s denotes the shadow value and c denotes the current value at the primary sever.

5) Determination of transaction type (Linear or Non-Linear)

To determine if the transaction is linear or non-liner, we need the program at the mobile unit to be more intelligent and performs more operations than data entry validation. The mobile unit determines the type of the transaction based on the mathematical function that is used to calculate the new value of the change aware attribute.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

TABLE 2: SALES TRANSACTION.

Read-Phase: K, A, R_o, W_o	10 , abc, 25 , 800
Edit-Phase: K, A, R_o, G, $F_1(R_o,g)$ ➔ $\Delta_{(W)}$ $\Delta_{(W)} + W_o = W_s$	10, abc, 25, 3 $F_1(25,3)$ ➔ -50 800 – 50 = 750 10,abc,25, 3, 750
Validation and Write Phase: Current Value at Primary Site: K, A, R_c, W_c	 10, abc, 25, 600

Validation Test:
If R_c <> R_o then
Rollback (t)
Else
$\Delta_{(W)} = W_s – W_o$
$W_c = W_c + \Delta_{(W)}$
 	
If(check-constraints(W_c) then
Accept W_c ,G
Commit (t)
Else
Rollback (t)
End if
End if
	25 : 25
	
	-50= 750 -800
	550 = 600-50
	
	check-constraints(550)= True
	Accept 550 , 3 , F(25,3) , -50
	Commit (t)

If the mathematical function includes only addition and/or subtraction operators (+, -), then the transaction is linear otherwise it is non-linear. The following function, we called it token-analysis, performs this analysis. The mobile unit passes the result of the analysis to the server as a parameter.

```
Function token_analysis(string fx , int tf)
{  char ch,  token [80];  int I= 0 ;
  While (not end of string fx) do
{
  Ch = getchar(fx)
If  ch != ' ' then
   Token [i] = ch;
        I++;
Else
If  token  = ( '^ '  or  '/ '  or '*'  or 'log' or 'len' or 'sin' or 'cos' or 'tan' ) then
   tf = 1        /*   which means non linear function
   Else
tf  = 0    /*  which means linear function
   End if
End if
}              /* while loop
Return (tf)  }
```

The validation-write procedure will be as follows:

Validation-Write-Procedure (Record original, Record shadow, String read-query, String update-query, integer Lflag)

```
Aware-Update (integer flag)
{
 For each change-reject-attribute(i)  in shadow-rec
  If  Current.R(i) <>  Shadow.R(i)    then
       Flag = -1
       Goto par-out
  End if
Next-For
For each change-aware-attribute(i)   in shadow-rec
 If   Current.W(i) =  Original.W (i)    then
Current.W(i)  = shadow(W(i))
 Else
If Lflag = 0 then
ΔW(i)   = Shadow.W(i)   - Original.W (i)
    Current.W(i)   = ΔW(i)   +  Current.W(i)
  Else
          Return (current.W(i))
    Call non-linear(current.w(i))
    Goto par-out
End if
 End if
   If (check-constraints(current.W(i)   ) = False ) then
         Flag = -2
         Goto par-out
    End if
Next-For
  Par-out:
 Return (flag)   }
```

Table 3 shows an example that explains how the M-Shadow technique handles both linear and non-linear transactions.

In table 3, T1 is applied as a linear transaction and T2 as a non-linear transaction. Both transactions decrease the value of the change-aware attribute (X) by 40 units, and both cases recalculate the new value (X). In the linear case, the recalculation is done by applying Δ(x). But in the non-linear case, the recalculation is done by passing the current value (X) at the primary server to the mobile unit which applies the mathematical function of the application that uses to calculate the new value of the change aware attribute (X). Both cases, linear and non-linear are serializable. For simplicity in the previous example, we used f(X) = X* 8/10, but it can be any function of the non-linear functions as f(X) = Xn or f(X) = log(X), etc. The logic that has been

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

applied on this function can be applied using any other non-linear function.

TABLE 3: LINEAR AND NON-LINEAR TRANSACTIONS.

T_1 (Linear)	T_2(Non-Linear)
Read Phase: Xoriginal = 200, X >= 0 **Edit Phase:** Xshadow = X-40= 160 **Validation Phase:** Xcurrent = 50 Δ(x) = Xshadow – Xoriginal - 40 = 160 - 200 Xcurrent = Xcurrent + Δ(x) 10 = 50 -40 Check-Constrains (10) Commit	**Read Phase:** Xoriginal = 200, X >= 0 **Edit Phase :** Xshadow = X* 8/10 = 160 **Validation Phase:** Xcurrent = 50 Send (Xcurrent) Accept (new-Xcurrent) from the client-agent (50 *8/10= 40) Check-Constrains(40) Commit

Figure (2) summarizes the M-Shadow model in a graphical representation. It shows the transaction processing between the mobile unit and the primary server. The application starts by asking the mobile user if the new transaction is location dependent or independent. If it is location dependent, the mobile unit sends a query and receives the response from the GPS system. Then the mobile user starts the edit phase as location independent transaction. The application determines if the transaction uses liner or non-liner functions.

The primary server performs the validation phase under exclusive-lock. If the data does not change at the primary server, it accepts the shadow data and copies it as a current data. If the data at the primary server changed, it checks the linear-flag, if its value is 0, which means linear transaction; it calculates the new value of the change aware attribute by calculating the difference between shadow and original values and add it to the current. If linear-flag is 1, which means non-linear transaction, it returns the current-value of the change aware attribute to the mobile unit to be recalculated at the mobile unit and re-passed to the server after recalculation. In this case, the new value of the change-aware attribute will be validated that it doesn't violates the enterprise constrains only, because validation phase is under exclusive-lock.

6) Advantages and Limitations of the M-Shadow technique

The advantages of using the shadow technique in general and the M-Shadow technique are:

1. Increase the performance of the system, by increasing the success probability of transaction by allowing transaction to continue its work even after disconnection and changing data on the primary server.
2. No transfer of logs or transaction history among sites. Only external files (XML files) would be saved on the mobile unit and will be deleted when the transaction finished.
3. Recovery for active transactions at failure time, which DBMS recovery manager does not do.
4. Decrease the programming time for applications, because the DBMS performs the update process.
5. No need to load the mobile unit with DBMS, replica and synchronization of replica.
6. No storage lost on the primary server or on the mobile unit, because after the transaction committed or roll backed, the program deletes the XML files.
7. The load on the primary server would be more lite.
8. More control over the network disconnection, especially in wireless networks which its property is frequently disconnection.
9. All ACID properties are reserved in the dependent case, and semantic ACID properties are reserved in the independent case.
10. This technique decreases the deadlock rate, or approximately, avoids the deadlock problem, because the locking of data-items at the primary server is very short and does not use shared lock.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

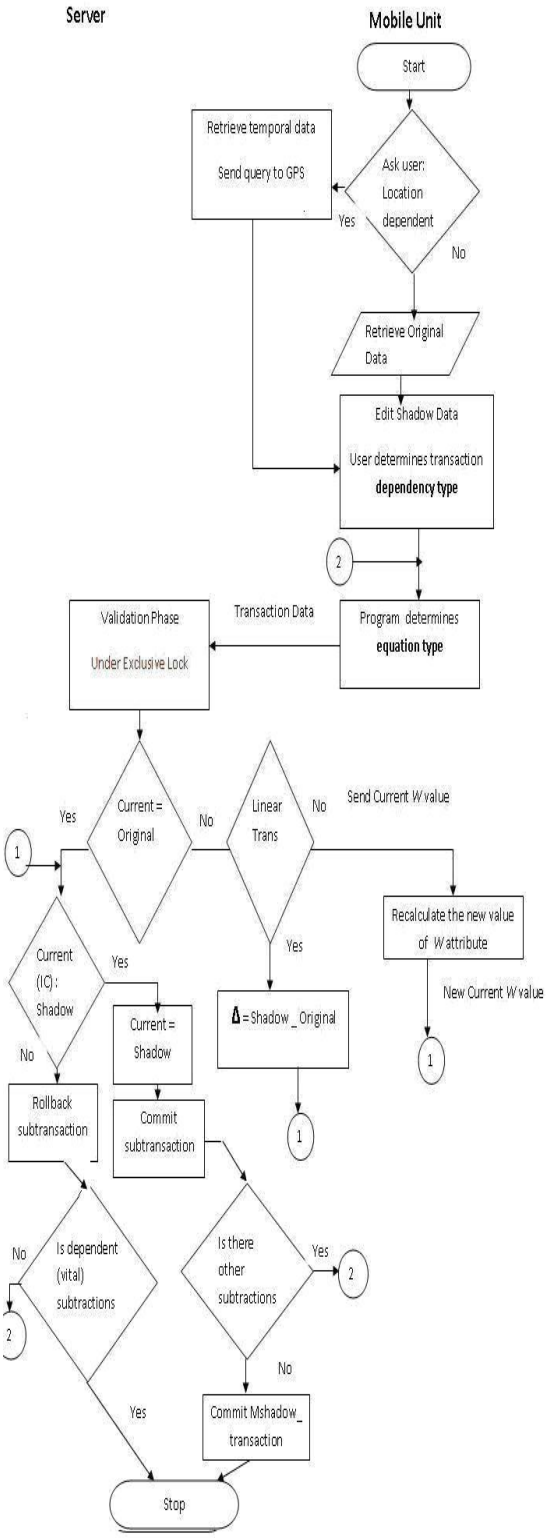

Figure 2 M-Shadow transaction model summary

Therefore, it increases the performance of the system.

11. It supports spatial databases transaction processing.

The limitations of the M-Shadow technique are: it is designed for commercial applications that have a few shared data-items among transactions and the validation test is not suitable for some real-time applications.

IV. SUMMARY OF IMPLEMENTATION AND PERFORMANCE EVALUATION

To evaluate the effects of using the actionability types and rules, we used the simulation program Benchmark Factory for Databases, but it does not allow to change data while the simulation process is running. So that, we developed the M-Shadow model with and without actionability. We found that, in the independent case and without actionability, the transaction that fails because of any data change at the primary server; it succeeds when the actionability types and rules are applied, that increases the number of succeed transactions and the success rate.

Also, in the dependent and in partially dependent cases and without actionability, the group transaction that fails if one of its vital subtransactions fails because of any data change at the primary server; it succeeds when the actionability types and rules are applied. So that, using the actionability types and rules increase the performance of the system by decreasing the number of aborted transactions.

We implemented a sales application that uses the M-Shadow technique as a location independent case using Visual Basic .Net and SQL Server 2005 because they support many new features as writing and reading XML files. We assume that the replication handling is solved as a distributed database problem using the lazy replication technique among fixed hosts.n

V. CONCLUSION AND FUTURE WORKS

In M-Shadow we increase the transaction success probability, this by consequence, raises the performance of the system. Actionability classifies the data elements handled by a transaction according to how much a change on these elements affects the transaction behaviour. 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. Finally, we described why validation and write phase should be run under exclusive lock.

Future research will extend this work to support complex business applications that include a big number of shared data items and complex computations, dependency among group transactions, parallel processing and real-time environments. Also, we will study how to find the optimal solution for selecting the next server in a shared area among many servers to decrease the number of disconnection. In

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

addition, security of mobile transactions will be investigated.

REFERENCES

[1] Romani Farid Ibrahim, *Handling Disconnection in Mobile Database Transaction*, The 5th International Conference on Application of Information nd Communication Technologies
[1] (AICT2011), Azerbaijan, Baku, 2011.
[2] Osman Hegazy, Ali El Bastawissy and Romani Farid Ibrahim, *Handling Mobile Transactions with Disconnections using a Mobile-Shadow Technique*, Proceedings of the 6th International conference of Informatics and Systems (INFOS 2008), Faculty of Computers & Information-Cairo University, March 2008.
[3] Osman Hegazy, Ali El Bastawissy and Romani Ibrahim, *A Programming Solution for Moving Mobile transaction*, Proceedings of the 6th International Enformatika (IEC 2005), Budapest, Hungary, October 2005.
[4] Osman Hegazy, Ali El Bastawissy and Romani Ibrahim, *Technique for Handling Transactions that Move among Hosts in Mobile Databases*, Proceedings of the International Conference on Computational Intelligence (ICCI 2004), Istanbul, Turkey December 2004.
[5] M. Dunham and A. Helal, *A Mobile Transaction Model that Captures both the Data and Movement Behaviour*, Mobile Networks and Application (MONET), pp149–162, 1997.
[6] Gary D. Walborn and Panos K. Chrysanthis, *PRO-MOTION: Management of mobile transactions*, Proceedings of ACM symposium on Applied computing April 1997.
[7] P. K. Chrysanthis, *Transaction Processing in a Mobile Computing Environment*, Proceedings of the IEEE Worskhop on Advances in Parallel and Distributed Systems, 1993.
[8] Yin-Huei Loh, Takahiro Hara, Masahiko Tsukamoto, Shojiro Nishio, *Moflex Transaction Model for Mobile Heterogeneous Multidatabase Systems*, Proceedings of the symposium on Applied computing, ACM, March 2000.
[9] Nadia Nouali, Anne Doucet and Habiba Drias, *A Two-Phase Commit Protocol for Mobile Wireless Environment*, Proceedings of the 16th Australasian Database Conference (ADC 2005), Australian Computer Society, vol. 39, pp135–143, 2005.
[10] A. Elmagarmid, J. Jing, J. G. Mullen and J. Sharif-Askary, *Reservable Transactions: An Approach for Reliable Multidatabase Transaction Management*, Technical Report SERC-TR-114-P, Software Engineering Research Centre, April 1992.
[11] S. Mazumdar and P.K. Chrysanthis, *Achieving Consistency in Mobile Databases through Localization in PRO-MOTION*, Proceedings of Int'l Conference and Workshop Database and Expert Systems Applications, (DEXA), IEEE, 1999.
[12] Tim Edmonds, Steve Hodges and Andy Hopper, *Pervasive Adaptation for Mobile Computing*, Proceedings of the 15th International Conference on Information Networking, 2001.
[13] M. Satyanarayanan, *Fundamental Challenges in Mobile Computing*, Proceedings of ACM Symposium on Principles of Distributed Computing, 1996.
[14] Thomas M.Connolly and Carolyn E.Begg, *Database Systems-A Practical Approach to Design, Implementation*, and Management", Addison Wesley, , pp 570-595, 2002.
[15] Jose Maria Monteiro, Angelo Brayner and Sergio Lifschitz, *A Mechanism for Replicated Data Consistency in Mobile Computing Environments*, Proceedings of ACM symposium on Applied computing, Seoul, Korea, pp 914-919, March, 2007.
[16] Vijay Kumar, *Mobile Database Systems*, John Wiley & Sons, pp 113- 197, 2006.

Biography

Romani Ibrahim received the B.A. in computer and information system from Sadat Academy for Management Science, Egypt. M.Sc. degree in computer science and Ph.D. degree in information systems from Cairo University, Egypt. He is an Assistant Professor in the department of computer information systems, Faculty of Information Technology, Al Hussein Bin Talal University, Jordan (by contract). In Egypt, he works in the High Institute of Computer Science and Information - City of Culture and Science- 6 October City. Egypt. He is a member of ACM. His research interests include distributed and mobile database systems, transaction processing, data warehousing and information security.