

**CERIAS Tech Report 2002-63**  
**Data and Transaction Management in a Mobile Environment**  
by S Madria, B Bhargava, M Mohania, S Bhowmick  
Center for Education and Research  
Information Assurance and Security  
Purdue University, West Lafayette, IN 47907-2086

## Chapter 8

# DATA AND TRANSACTION MANAGEMENT IN A MOBILE ENVIRONMENT

Sanjay Madria, Bharat Bhargava, Mukesh Mohania, Sourav Bhowmick  
*Department of Computer Science, University of Missouri-Rolla, USA; Department of Computer Sciences, Purdue University, USA; Department of Computer Science, Western Michigan University, USA; School of Computer Engineering, Nanyang Technological University, Singapore*

**Abstract:** The mobile computing paradigm has emerged due to advances in wireless or cellular networking technology. This rapidly expanding technology poses many challenging research problems in the area of mobile database systems. Mobile users can access information independent of their physical location through wireless connections. However, accessing and manipulating information without restricting users to specific locations complicates data processing activities. There are computing constraints that make mobile database processing different from the wired distributed database computing. In this chapter, we survey the fundamental research challenges particular to mobile database computing, review some of the proposed solutions and identify some of the upcoming research challenges. We discuss interesting research areas, which include mobile location data management, transaction processing and broadcast, cache management and replication. We highlight new upcoming research directions in mobile digital library, mobile data warehousing, mobile workflow and mobile web and e-commerce.

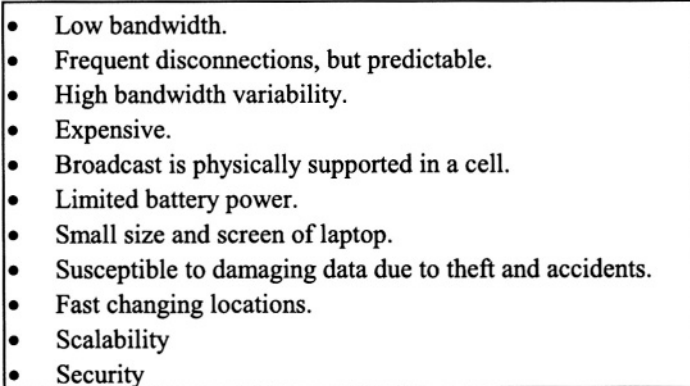
**Keywords:** Distributed database, Mobile computing, Mobile database, Wireless or cellular networking

## 1. INTRODUCTION

The rapid technological advancements in cellular communications, wireless LAN and satellite services have led to the emergence of mobile

computing [1]. In mobile computing, users are not attached to a fixed geographical location, instead their point of attachment to the network changes as they move. The emergence of relatively sophisticated low-power, low-cost and portable computing platforms such as laptops and personal digital assistants (PDA) have made it possible for people to work from anywhere at any time (from their offices, homes, and while travelling) via a wireless communication network to provide unrestricted user mobility.

Mobility and portability pose new challenges to mobile database management and distributed computing [2]. There is a necessity to design specifications for energy efficient data access methodologies and in general develop database software systems that extend existing database systems designs and platforms to satisfy the constraints imposed by mobile computing (see Figure 1). How to handle long periods of disconnection, and other constrained resources of mobile computing such as limited battery life and variable bandwidth etc.? In mobile computing, there will be more competition for shared data since it provides users with the ability to access information and services through wireless connections that can be retained even while the user is moving. Further, mobile users will have to share their data with others. The task of ensuring consistency of shared data becomes more difficult in mobile computing because of limitations and restrictions of wireless communication channels.

- 
- Low bandwidth.
  - Frequent disconnections, but predictable.
  - High bandwidth variability.
  - Expensive.
  - Broadcast is physically supported in a cell.
  - Limited battery power.
  - Small size and screen of laptop.
  - Susceptible to damaging data due to theft and accidents.
  - Fast changing locations.
  - Scalability
  - Security

*Figure 1. Constraints of mobile computing*

In this chapter, we discuss some of the problems identified with mobile database computing, review proposed solutions, and explore the upcoming research challenges.

The rest of the chapter is as follows: In Section 2, we discuss mobile database architecture. Section 3 contains mobile data management research issues. In Section 4, we discuss transaction management issues in mobile

computing. Section 5 presents some research directions in mobile data management, and the last section concludes the chapter.

## 2. MOBILE DATABASE ARCHITECTURE

In a mobile computing environment (see Figure 2), the network consists of Fixed Hosts (FH), Mobile Units (MU) and Base Stations (BS) or Mobile Support Stations (MSS). MUs are connected to the wired network components only through BS via wireless channels. MUs are battery powered portable computers, which move around freely in a restricted area, which we refer to as the "geographical region" (G). For example in Figure 2, G is the total area covered by all BS. This cell size restriction is mainly due to the limited bandwidth of wireless communication channels. To support the mobility of MUs and to exploit frequency reuse, the entire G is divided into smaller areas called cells. Each cell is managed by a particular BS. Each BS will store information such as user profile, log-in files, access rights together with user's private files. At any given instant, a MU communicates only with the BS responsible for its cell. The mobile discipline requires that a MU must have unrestricted movement within G (inter-cell movement) and must be able to access desired data from any cell.

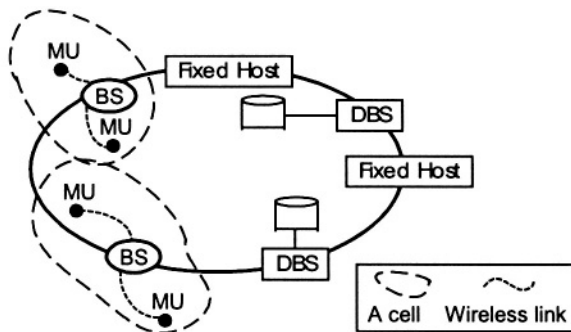


Figure 2. Architecture of MDS

A mobile unit (MU) changes its location and network connections while computations are being processed. While in motion, a mobile host retains its network connections through the support of base stations (BS) with wireless connections. The BSs and FHs (fixed hosts) perform the transaction and data management functions with the help of a database server (DBS) component to incorporate database processing capability without affecting any aspect of

the generic mobile network. DBSs can either be installed at BSs or can be a part of FHs or can be independent of BS or FH.

When a MU leaves a cell serviced by a BS, a hand-off protocol is used to transfer the responsibility for mobile transaction and data support to the BS of the new cell. This hand-off involves establishing a new communication link. It may also involve migration of in progress transactions and database states from one BS to another. The entire process of handoff is transparent to a MU and is responsible for maintaining end-to-end data movement connectivity.

## 2.1 Modes of Operations

In mobile computing, there are several possible modes of operations [3,4], whereas in a traditional distributed system, a host may operate only in one of two modes; either connected to the network or totally disconnected. The operation mode in mobile computing may be one of the following:

- fully connected (normal connection)
- totally disconnected (e.g., not a failure of MU)
- partially connected or weak connection (a terminal is connected to the rest of the network via low bandwidth).

In addition, for conserving energy, a mobile computer may also enter an energy conservation mode, called *doze state* [3]. A doze state of MU does not imply the failure of the disconnected machine. In this mode, the clock speed is reduced and no user computation is performed.

Most of these disconnected modes are predictable in mobile computing. Protocols can be designed to prepare the system for transitions between various modes. A mobile host should be able to operate autonomously even during total disconnection.

A *disconnection protocol* is executed before the mobile host is physically detached from the network. The protocol should ensure that enough information is locally available (cached) to the mobile host for its autonomous operation during disconnection. It should inform the interested parties about the forthcoming disconnection.

A *partially-disconnection protocol* prepares the mobile host for operation in a mode where all communication with the fixed network is restricted. Selective caching of data at the host site will minimize future network use.

*Recovery protocols* re-establish the connection with the fixed network and resume normal operation.

Similar terminology has been used later.

### **3. MOBILE DATA MANAGEMENT**

In this section, we will discuss some of the important data management issues with respect to mobile computing.

#### **3.1 Location Data Management**

The location of a mobile user is of prime importance in wireless computing. In mobile computing, the location of a user can be regarded as a data item whose value changes with every move. In mobile computing, the location management is a data management problem; that is, the managing location data encounters the same problems as in managing normal data. Primary issues here are how to know the current position of the mobile unit. Where to store the location information and who should be responsible for determining and updating of information? To locate users, distributed location databases are deployed which maintain the current location of mobile users. The location data can be treated as a piece of data that is updated and queried. The search of this piece of data should be as efficient as any other queried data. Writing the location variable may involve updating the location of the user in the location database as well as in other replicated databases. The location management involves, searching, reading, informing and updating. If A wants to find the location of B, should A search the whole network or only look at pre-defined locations? Should B inform any one before relocating? One such method is described in [5]. It assumes that each user is attached to a home location server (now generally referred as home location register (HLR)) that always “knows” his current address. When a user moves, he informs his home location server about his new address. To send a message to such a user, his home location register is contacted first to obtain his current address. A special form of “address embedding” is used to redirect the packets addressed to the mobile user from the home location to his current location. This scheme works well for the user who stays within their respective home areas, it does not work for global moves. In this algorithm, when a user A calls user B, the lookup algorithm initiates a remote lookup query to the HLR of B, which may be at remote site. Performing remote queries can be slow due to high network latency. An improvement over such algorithm [6] is to maintain visitor location registers (VLR). The VLR at a geographical area stores the profiles of users currently located in that area for whom the area is not their home. The query then calls in the caller’s area and if the callee’s profile is not found, it queries the database in the callee’s home area. This is useful when a callee received many calls from users in the area he is visiting since it avoids queries to the HLR of the callee at the remote site. VLR’s can be viewed as a

limited replication scheme since each user's profile is located in its current area when he is not in his home area. Another scheme proposed in [7], handles global moves on the assumption that most messages are exchanged between parties or between users in a remote area and its home location area.

A formal model for online tracking of users is considered in [8] by decomposing a PCS (Personal Communication System) network into regions and using regional directories. The authors discuss how to trade-off search and update costs while tracking users. Authors in [9,10] propose per-user placement, which uses cell partitions where the user travels frequently and separating the cells between which it relocates infrequently to control network traffic generated by frequent updates. Only moves which are across the partitions, are reported.

In [11], the location lookup problem is considered to find a callee within the reasonable time bounds to set up the call from the caller to callee. Each user is located in some geographical area where the mobile service station keeps track of each user in the form of  $\langle \text{PID}, \text{ZID} \rangle$  where PID and ZID uniquely identified the mobile unit id and its current location id, respectively. They replicate per-user profiles based on calling and mobility patterns. Thus, they balance the storage and update costs and at the same time provide fast lookups. The decision of where to replicate the profiles is based on a minimum-cost maximum-flow [12] algorithm. They maintain an up-to-date copy of a user profile at the user's HLR and in addition, they also find out the sites at which a user's profile will be replicated. Thus, the algorithm does not guarantee that a user's profile will be found in his current area.

Hierarchical distributed database architectures [13,14,15,16,17] have been built to accommodate the increased traffic associated with locating moving users. In these models, each leaf database covers a specific geographical region and maintains location information for all users currently residing in that region. Location databases at internal nodes contain information about the location of all the users registered in areas covered by the databases at their children nodes. A hierarchical method for location binding in a wide-area system is used in the Globe wide-area location service [14]. Globe uses a combination of caching and partitioning.

A tree based structure is used for a location database in [18]. The authors modify the structure to balance the average load of search requests by replacing the root and some of the higher levels of the tree with a set-ary butterfly (a generalization of K-ary butterfly). They modify the lowest level of the tree to reflect neighbouring geographical regions more accurately and to allow simple hand-offs.

In [19], the hierarchical scheme allows dynamic adjustment of the user location information distribution, based on mobility patterns of mobile units.

A unique distribution strategy is determined for each mobile terminal and location pointers are set up at selected remote locations. This reduces database access overhead for registration and there is no need for centralized co-ordination.

Forwarding pointers have been used in hierarchical location databases [15,17]. In [15], the objective is to reduce the cost of moves by updating only databases up to a specific level of the tree and a forwarding pointer is set at a lower level in the database. However, if forwarding pointers are never deleted, then long chains are created, whose traversal results increases the cost of locating users during calls. They introduced caching techniques that reduce the number of forwarding pointers to travel before locating the calling as well as conditions for initiating a full update of the database entries. They have also described a synchronization method to control the concurrent execution of call and move operations. The difference between the two schemes [15,17] is that in [17] the actual location is saved at each internal level database instead of pointer to the corresponding lower level database. The forwarding pointers are set at different levels in the hierarchy and not necessarily at the lower level database as in [15]. In [17], the objective is to choose an appropriate level for setting the forwarding pointers and on updating obsolete entries in the hierarchy after a successful call. Caching in hierarchical structures is proposed in [13] instead of replication to reduce the cost of calls.

### **3.2 Cache Consistency**

Caching of frequently accessed data plays an important role in mobile computing because of its ability to alleviate the performance and availability limitations during weak-connections and disconnections. Caching is useful during frequent relocation and connection to different database servers. In wireless computing, caching of frequently accessed data items is an important technique that will reduce contention on the small bandwidth wireless network. This will improve query response time, and to support disconnected or weakly connected operations. If a mobile user has cached a portion of the shared data, he may request different levels of cache consistency. In a strongly connected mode, the user may want the current values of the database items belonging to his cache. During weak connections, the user may require weak consistency when the cached copy is a quasicopy of the database items. Each type of connection may have a different degree of cache consistency associated with it. That is, a weak connection corresponds to a “weaker” level of consistency.

Cache consistency is severely hampered by both the disconnection and mobility of clients since a server may be unaware of the current locations



and connection status of clients. This problem can be solved by the server by periodically broadcasting either the actual data, invalidation report (reports the data items which have been changed), or even control information such as lock tables or logs. This approach is attractive in mobile environments since the server need not know the location and connection status of its clients and clients need not establish an up link connection to a server to invalidate their caches. There are two advantages of broadcasting. First, the mobile host saves energy since they need not transmit data requests and second, broadcast data can be received by many mobile hosts at once with no extra cost.

Depending upon what is broadcasted, the appropriate schemes can be developed for maintaining consistency of data of a distributed system with a mobile client. Given the rate of updates, the trade-off is between the periodicity of broadcast and divergence of the cached copies that can be tolerated. The more the inconsistency is tolerated the less often the updates need to be broadcasted. Given a query, the mobile host may optimise energy costs by determining whether it can process the query using cached data or transmit a request for data. Another choice could be to wait for the relevant broadcast.

Cache coherence preservation under weak-connections is expensive. Large communication delays increase the cost of validation of cached objects. Unexpected failures increase the frequency of validation since it must be performed each time communication is restored. An approach that only validates on demand could reduce validation frequency but this approach would worsen consistency since it increases the possibility of some old objects being accesses while disconnected.

In Coda [20], during the disconnected operation, a client continues to have read and write access to data in its cache. The Coda file system allows the cached objects in the mobile host to be updated without any co-ordination. When connectivity is restored, the system propagates the modifications and detects update conflicts. The central idea is that caching of data and the key mechanisms for supporting disconnected operations which includes three states: hoarding, emulation and reintegration. The client cache manager while in hoarding state relies on server replication, but is always on the alert for possible disconnection and ensures that critical objects are cached at the time of disconnection. Upon disconnection, it enters the emulation state and relies solely on the contents of the cache. Coda's original technique for cache coherence while connected was based on callbacks [21]. In this technique, a server remembers that a client has cached an object, and promises to notify it when another client updates the object. This promise is called callback, and the invalidation message is a callback break. When a callback break is received, the client discards the cached copy and refetches

it on demand. When a client is disconnected, it can no longer rely on callbacks. Upon reconnection, it must revalidate all cached objects before use to detect updates at the server.

Cache invalidation strategies will be affected by the disconnection and mobility of clients. The server may not have information about the live mobile units in its cells. [22] proposes taxonomy of different cache invalidation schemes and studies the impact of a client's disconnection times on their performance. They address the issue of relaxing consistency of caches. They use quasi-copies whose values can deviate in a controlled manner. They have categorized the mobile units on the basis of the amount of time they spend in their sleep mode into sleepers, and workaholics. Different caching schemes turn out to be effective for different populations. Broadcast with timestamps are proved to be advantageous for frequent queries than the rate of updates provided that units are not workaholics.

A technique is proposed in [23] to decide whether the mobile unit can still use some items in the cache even after it is connected to the server. The database is partitioned into different groups and items in the same group are cached together to decrease the traffic. Thus, a mobile unit has to invalidate only the group rather than individual items. In [24], various alternative caching strategies for mobile computing have been evaluated. More work needs to be done in the direction of performance evaluation and the availability limitation of various caching under weakly-connected and disconnected operation.

In [25], an incremental cache coherency problem in mobile computing is examined in the context of relational operations select, project and join. A taxonomy of cache coherency schemes is proposed as case studies. However, it does not address the problems of query processing and optimisation and does not include other relational operations.

### **3.3 Data Replication**

The ability to replicate the data objects is essential in mobile computing to increase availability and performance. Shared data items have different synchronization constraints depending on their semantics and particular use. These constraints should be enforced on an individual basis. Replicated systems need to provide support for disconnected mode, data divergence, application defined reconciliation procedures, optimistic concurrency control, etc. Replication is a way by which the system ensures transparency for mobile users. A user who has relocated and has been using certain files and services at the previous location wants to have his environment recreated at the new location. Mobility of users and services and its impact on data replication and migration will be one of the main technical problems to be

resolved. There are many issues raised by the relocated data and mobility of users and services:

- How to manage data replication, providing the levels of consistency, durability and availability needed?
- How to locate objects of interest? Should information about location be also replicated and to what extent (location is dynamically changing data item)?
- What are the conditions under which we need to replicate the data on a mobile site?
- How users' moves affect the replication scheme. How should the copy follow the user? In general data should move closer to the user?
- Is a mobile environment requiring dynamic replication schemes [26]?
- Do we need new replication algorithms or the proposed replication schemes for distributed environment can be modified?

In [27], caching of data in mobile hosts and the cost of maintaining consistency among replicated data copies have been discussed. It allows caching of data to take place anywhere along the path between mobile/fixed servers and clients. It determines, via simulations, which caching policy best suits given mobility and read/write patterns.

A general model is considered in [28] for maintaining consistency of replicated data in distributed applications. A casualty constraint, a partial ordering between application operations, is defined such that data sharing is achieved by defining groups requiring it and broadcasting updates to the group. Each node processes the data according to the constraints.

In [29], it has been argued that traditional replica control methods are not suitable for mobile databases and the authors have presented a virtual primary copy method. In this method, the replica control method decides on a transaction-by-transaction basis whether to execute that transaction on a mobile host's primary copy or virtual primary copy. This method requires a transaction to be restarted when a mobile host disconnects. Also, when a mobile host reconnects, it either has to wait for the completion of all transactions executed on a virtual copy before synchronizing itself with the rest of the system or all running transactions will have to be restarted.

An analysis of various static and dynamic data allocation methods is presented in [30] with the objective of optimising the communication cost between a mobile computer and the stationary computer that stores an online database. The authors consider one-copy and two-copies allocation schemes. In the static scheme, an allocation remains unchanged where as in a dynamic scheme allocation method changes are based on the number of reads and writes. If in the last  $k$  requests there are more reads at an MU than writes at a stationary computer, it uses the two-copy scheme. Otherwise it uses one-copy schemes. Two costs models were developed for cellular phones (user is

charged per minute of connection) and packet radio networks (user is charged per message basis), respectively.

A new two-tier replication algorithm is proposed by Gray et al. in [31] to alleviate the unstable behaviour observed in the update anywhere-anytime-anyway transactional replication scheme when the workload scales up. Lazy master replication that is employed in the algorithm assigns an owner to each object. The owner stores the object's correct value. Updates are first done by the owner and then propagated to other replicas. The two tier scheme uses two kinds of nodes: mobile nodes (may be disconnected) and base nodes (always connected). The mobile nodes accumulate tentative transactions that run against the tentative database stored at the node. Each object is mastered at either the mobile node or the base node. When the mobile node reconnects to the base station, it sends replica updates mastered at the mobile node, the tentative transactions and their input parameters to the base node. They are to be re-executed as base transactions on the master version of data objects maintained at the base node in the order in which they are committed on the mobile node. If the base transaction fails its acceptance criterion, the base transaction is aborted and a message is returned to the user of the mobile node. While the transaction executed on the objects mastered on the mobile nodes are confirmed, those executed on the tentative objects have to be checked with nodes that hold the master version.

A dynamic replication scheme which employs user profiles for recording users' mobility pattern, access behaviour and read/write patterns, also actively reconfigures the replicas to adjust to the changes in the user's locations and systems is proposed in [32]. They devise the concept of open objects to represent a user's current and near future data requirements. This leads to a more precise and responsive cost model to reflect changes in access patterns.

#### **4. MOBILE TRANSACTION PROCESSING**

A transaction in mobile environment is different than the transactions in the centralised or distributed databases in the following ways:

- The mobile transactions might have to split their computations into sets of operations, some of which execute on a mobile host while others execute on stationary host.
- A mobile transaction shares its states and partial results with other transactions due to disconnection and mobility.
- The mobile transactions require computations and communications to be supported by stationary hosts.

- When the mobile user moves during the execution of a transaction, it continues its execution in the new cell. The partially executed transaction may be continued at the fixed local host according to the instruction given by the mobile user. Different mechanisms are required if the user wants to continue the transaction at a new destination.
- As the mobile hosts move from one cell to another, the states of transaction, states of accessed data objects, and the location information also move.
- The mobile transactions are long-lived transactions due to the mobility of both the data and users, and due to the frequent disconnections.
- The mobile transactions should support and handle concurrency, recovery, disconnection and mutual consistency of the replicated data objects.

To support mobile transactions, the transaction processing models should accommodate the limitations of mobile computing, such as unreliable communication, limited battery life, low band-width communication, and reduced storage capacity. Mobile computations should minimise aborts due to disconnection. Operations on shared data must ensure correctness of transactions executed on both the stationary and mobile hosts. The blocking of a transaction's executions on either the stationary or mobile hosts must be minimized to reduce communication cost and to increase concurrency. Proper support for mobile transactions must provide for local autonomy to allow transactions to be processed and committed on the mobile host despite temporary disconnection.

Semantic based transaction processing models [33,34] have been extended for mobile computing in [35] to increase concurrency by exploiting commutative operations. These techniques require caching a large portion of the database or maintain multiple copies of many data items. In [35], fragmentability of data objects have been used to facilitate semantic based transaction processing in mobile databases. The scheme fragments data objects. Each fragmented data object has to be cached independently and manipulated synchronously. That is, on request, a fragment of a data object is dispatched to the MU. On completion of the transaction, the mobile hosts return the fragments to the BS. Fragments are then integrated in the object in any order and such objects are termed as reorderable objects. This scheme works only in the situations where the data objects can be fragmented like sets, stacks and queues.

In optimistic concurrency control based schemes [36], cached objects on mobile hosts can be updated without any co-ordination but the updates need to be propagated and validated at the database servers for the commitment of transactions. This scheme leads to aborts of mobile transactions unless the conflicts are rare. Since mobile transactions are expected to be long-lived

due to disconnection and long network delays, the conflicts will be more in a mobile computing environment.

In pessimistic schemes in which cached objects can be locked exclusively, mobile transactions can be committed locally. The pessimistic schemes lead to unnecessary transaction blocking since mobile hosts can not release any cached objects while it is disconnected. Existing caching methods attempt to cache the entire data objects or in some case the complete file. Caching of these potentially large objects over low bandwidth communication channels can result in wireless network congestion and high communication cost. The limited memory size of the MU allows for only a small number of objects to be cached at any given time.

Dynamic object clustering has been proposed in mobile computing in [4]. It assumes a fully distributed system, and the transaction model is designed to maintain the consistency of the database. The model uses weak-read, weak-write, strict-read and strict-write. The decomposition of operations is done based on the consistency requirement. Strict-read and strict-write have the same semantics as normal read and write operations invoked by transactions satisfying ACID (Atomicity, Consistency, Isolation and Durability) properties. A weak-read returns the value of a locally cached object written by a strict-write or a weak-write. A weak-write operation only updates a locally cached object, which might become permanent on cluster merging if the weak-write does not conflict with any strict-read or strict-write operation. The weak transactions use local and global commits. The local commit is the same as pre-commit of [37] and global commit is the same as a final commit in [37]. However, a weak transaction after local commit can abort and is compensated. In [37], a pre-committed transaction does not abort, hence requires no undo or compensation. A weak transaction's updates are visible to other weak transactions whereas prewrites are visible to all transactions.

A new transaction model using isolation-only transactions (IOT) is presented in [38]. The model supports a variety of mechanisms for automatic conflict detection and resolution. IOTs are sequences of file accesses that unlike traditional transactions have only isolation property. Transaction execution is performed entirely on the client and no partial result is visible on the servers. IOTs do not provide failure atomicity, and only conditionally guarantee permanence. They are similar to the weak transactions of [3].

An open nested transaction model has been proposed in [39] for modelling mobile transactions as a set of subtransactions. They introduce reporting and co-transactions. A reporting transaction can share its partial results, can execute concurrently and can commit independently. Co-transactions are like co-routines and are not executed concurrently. The model allows transactions to be executed on disconnection. It also supports

unilateral commitment of subtransactions, compensating and non-compensating transactions. The author claims that the model minimizes wired as well as wireless communication cost. However, not all the operations are compensated [39], and compensation is costly in mobile computing.

Transaction models for mobile computing that perform updates at mobile computers have been developed in [3,39]. These efforts propose a new correctness criterion [39] that is weaker than serializability. They can cope more efficiently with the restrictions of mobile and wireless communications.

In [37,40,41], we look at a mobile transaction more as a concurrency control problem and provide database consistency. We incorporate a prewrite operation [42] before a write operation in a mobile transaction to improve data availability. A prewrite operation does not update the state of a data object but only makes visible the value that the data object will have after the commit of the transaction. Once a transaction has received all the values read and declares all the prewrites, it can pre-commit at the mobile host (i.e., computer connected to unreliable communication) and the remaining transaction's execution is shifted to the stationary host (i.e., computer connected to the reliable fixed network). Writes on a database, after pre-commit, take time and resources at stationary host and are therefore, delayed. This reduces network traffic congestion. A pre-committed transaction's prewrite values are made visible both at the mobile and stationary hosts before the final commit of the transaction. This increases data availability during frequent disconnections that are common in mobile computing. Since the expensive part of the transaction's execution is shifted to the stationary host, it reduces the computing expenses (e.g., battery, low bandwidth, memory etc.) at the mobile host. Since a pre-committed transaction does not abort, no undo recovery needs to be performed in our model. A mobile host can cache only prewrite values of the data objects, which will take less space, time, and energy and can be transmitted over low bandwidth.

A kangaroo transaction (KT) model was given in [43]. It incorporates the property that transactions in a mobile computing hop from a base station to another as the mobile unit moves. The mobility of the transaction model is captured by the use of split transaction [44]. A split transaction divides on going transactions into serializable subtransactions. An earlier created subtransaction is committed and the second subtransaction continues its execution. The mobile transaction splits when a hop occurs. The model captures the data behaviour of the mobile transaction using global and local transactions. The model also relies on compensating transaction in case a transaction aborts. The model in [37] has the option of either using nested

transactions or split transactions. However, the save point or split point of a transaction is explicitly defined by the use of pre-commit. This feature of the model allows the split point to occur in any of the cells. Unlike the KT model, the earlier subtransaction after pre-commit can still continue its execution with the new subtransaction since their commit orders in the model [37] are based on the pre-commit point. Unlike the KT, the model in [37] does not need any compensatory transaction.

In [45], a basic architectural framework to support transaction management in multidatabase systems is proposed and discussed. A simple message and queuing facility is suggested which provides a common communication and data exchange protocol to effectively manage global transactions submitted by mobile workstations. The state of global transactions is modelled through the use of subqueues. The proposed strategy allows a mobile workstation to submit global transactions and then disconnected itself from the network to perform some other tasks thereby increasing processing parallelism and independence.

A transaction management model for the mobile multidatabase is presented in [46], called Toggle Transaction Management Technique. Here site transactions are allowed to commit independently and resources are released in timely manner. A toggle operation is used to minimize the ill-effects of the prolonged execution of long-lived transaction.

In most of the above papers, there is no comparative performance evaluation of models presented. We observe that there is a need to investigate the properties of mobility by means of experiments, which can impact most the transaction processing. Also, there is a need to evaluate various transaction processing algorithms with respect to performance, response time, throughput and may be a new paradigm like quality of service (QoS) in the transaction management in an area such as e-commerce.

## **4.1 Broadcast Disk and Transaction processing**

In traditional client-server systems, data are delivered from servers to clients on demand. This form of data is called pull-based. Another interesting trend is push-based delivery in a wireless environment. In wireless computing, the stationary server machines are provided with a relative high bandwidth channel that supports broadcast delivery to all mobile clients located inside the cell. In a push-based delivery, a server repetitively broadcasts data to clients without a specific request. Clients monitor the broadcast and retrieve data items they need as they arrive on the broadcast channel. This is very important for a wide range of applications that involve dissemination of information to a large number of clients. Such applications include stock quotes, mailing lists, electronic newsletters, etc.



Broadcast in a mobile computing has a number of difficulties. How to predict and decide about the relevance of the data to be broadcast to clients? One way is that the clients may subscribe their interests to services [17]. The server also needs to decide about either sending the data periodically [48] or aperiodically. Mobile clients are also resource poor and the communication environment is asymmetric. The problem also is to maintain the consistency of broadcast data. The commercial system which supports the concept of broadcast data delivery has been proposed in [49]. Recently, broadcast has received considerable attention in the area of transaction processing in mobile computing environments.

Pitoura et al. [50] addresses the problem of ensuring consistency and currency of client read-only transactions when the values are being broadcast in the presence of updates at the server. The authors propose to broadcast additional control information in the form of invalidation reports, multiple versions per item and serializability information. They propose different methods that vary in complexity and volume of control information transmitted and subsequently differs in response times, degree of concurrency, and space and processing overheads. The proposed methods are combined with caching to improve query latency.

Authors in [51] exploit versions to increase concurrency of client read-only transactions in the presence of updates at the servers. Invalidation reports are used to ensure the currency of reads. They broadcast older versions along with new values. The approach is scalable as it is independent of the number of clients. Performance results show that the overhead of maintaining older versions is low and at the same time concurrency increases. On the same line, [52] presents an approach for concurrency control in broadcast environments. They propose a weaker notion of consistency, called update consistency, which still satisfy the mutual consistency of the data.

Transactions support in a mobile database environment with the use of a broadcast facility is reported in [53]. Mobile clients use broadcast data to verify if transactions are serializable. [54] presents an optimistic concurrency control protocol with update timestamps to support transactional cache consistency in a wireless mobile computing environment using broadcast. They implement the consistency check on accessed data and the commitment protocol in a truly distributed fashion as a part of a cache invalidation process with most of the burden of consistency check being downloaded to mobile clients. They achieve improved transaction throughput in comparison to [53] and it minimizes the wireless communication required for supporting mobile transactions.

## **5. MOBILE DATABASE RESEARCH DIRECTIONS**

We see the following as upcoming mobile database research directions:

### **Location-dependent Query Processing**

We present ideas for processing queries that deal with location-dependent data [55]. Such queries we refer to as location-dependent. Location can be a subject of more complex aggregate queries, such as finding the number of hotels in the area you are passing or looking for a mobile doctor closest to your present location. Hence, the location information is a frequently changing piece of data. The objective is to get the right data at different locations for processing a given query. The results returned in response to such queries should satisfy the location constraints with respect to the point of query origin, where the results are received, etc. We propose to build additional capabilities into the existing database systems to handle location-dependent data and queries.

We present some examples to recognize the problems of accessing correct data when the point of contact changes. Data may represent SSN (Social Security Number) of a person, or maiden name, or sales tax of a city. In one representation, the mapping of the data value and the object it represents is not subjected to any location constraints. For example, the value of the SSN of a person remains the same no matter from which location it is accessed. This is not true in the case of sales tax data. The value of the sales tax depends on the place where a sales query is executed. For example, the sales tax value of West Lafayette is governed by a different set of criteria than the sales tax of Boston. We can therefore, identify the type of data whose value depends on the set of criteria established by the location and another type of data, which is not subject to the constraints of a location. There is a third type of data that is sensitive to the point of query. We illustrate this data with the following example. Consider a commuter who is traveling in a taxi and initiates a query on his laptop to find the nearby hotels in the area of its current location. The answer to this query depends on the location of the origin of the query. Since the commuter is moving he may receive the result at a different location. Thus, the query results should correspond to the location where the result is received or to the point of the origin of the query. The difference in these two correct answers to the query depends on the location and not on the hotel. The answer to the query “find the cheapest hotel” is not affected by the movement. The former depends on the location whereas later depends on the object characteristics. [55] discusses the data organization issues in location dependent query processing.

In [56], queries with location constraints are considered. They consider the query such as “find the nearest hotel from my current position”. The main objective there is to minimize the communication cost to retrieve the necessary information to answer the query. The authors suggest greedy heuristics to solve the problem.

In the MOST project [57,58], the authors consider a database that represents information about moving objects and their location. They argue that existing DBMS's are not well equipped to handle continuously changing data, such as location of moving objects. They address the issue of location modelling by introducing the concepts of dynamic attribute (whose value keeps changing), spatial and temporal query languages and indexing dynamic attributes.

### **View Maintenance in Mobile Computing**

Accessing on-line database from a mobile computer may be expensive due to limited uplink bandwidth, and also due to the fact that sending messages consumes considerable energy which is limited in the portable battery. These two problems can be solved by maintaining a materialised view i.e., storing the tuples of the view in the database at the mobile computer. This view will be updated as the on-line database changes using wireless data messages. This will also localise access, thus improving access time. Therefore, to better deal with the problem of disconnection, reliability, and to improve response time, the view should be materialised at the mobile computer. The view maintenance will involve location-dependent data, time-dependent data, and dynamic allocation of a materialised view in the fixed and mobile network. Another problem is dynamic allocation of a materialized view in the mobile network [30]. More work in this direction has been reported in [59]. Another issue concerns the divergence of the materialised view at the mobile computer from the on-line database. In other words, how closely should the materialised view reflect the on-line database. An approach given in [60] involves parameterising each read at the mobile computer with the amount of divergence from the latest version it can accept. Another approach given in [61] allows the user to specify triggers on the on-line database and the view is updated when the trigger is satisfied. Recently, this problem has also been discussed in a position paper [62] to emphasise the importance of data warehousing in view maintenance in a mobile computing environment.

There is a need to develop view maintenance algorithms in the case of data warehousing environment where relational data is broadcast. Similarly, there is a need to do change management in the web data where changes are broadcast periodically and mobile host should capture the data and make the cached web data consistent.

## **Workflows in Mobile Environment**

Workflow management systems are growing due to their ability to improve the efficiency of an organization by streamlining and automating business processes. Workflow systems have to be integrated with the mobile computing environment [63] in order to co-ordinate a disconnected computer to enhance the system's resilience to failures. Specific issues that arise here include how the workflow models can co-ordinate tasks that are performed when mobile users work in disconnected mode and when they cross wireless boundaries. Also location sensitive activities might have to be scheduled to use an organization's resources effectively. Current workflow systems do not seem to have any provision to handle these requirements.

Authors in [63] discuss how disconnected workflow clients can be supported while preserving the correctness of the overall execution and allowing co-ordinated interactions between the different users. The required activities (i.e., applications and data) are downloaded onto the mobile computer before performing a planned disconnection. The activities are performed in a disconnected mode and the results are then uploaded after reconnection at which time exceptions that occurred during the disconnected mode of operations are also handled. When mobile users cross the borders of wireless cells, a hand-off (i.e., a mobile user's session is to migrate to a new information server) may need to be performed from the old workflow server to a new server. Consistency issues that arise when workflow instances migrate are to be handled carefully. For location sensitive routing of a mobile user's request, the modelling primitives should have a provision to specify geographic information in the workflow definition.

## **Mobile Web and E-commerce**

There is a need to bring the web onto a mobile platform. Imagine a taxi that is equipped with a mobile computer and a passenger would like to browse web pages while waiting to reach their destinations. The limited bandwidth will be a bottleneck in such a scenario. Another interesting application may be e-commerce on the mobile web. All these applications are viable for the disconnected, highly unreliable, limited bandwidth and unsecured platform, but where the application demands reliability and security. Some efforts in this direction have appeared in [64], which is a WWW system designed to handle mobile users. It allows the documents to refer and react to the current location of clients. In [65], they address the issue of mobile web browsing through a multi-resolution transmission paradigm. The multi-resolution scheme allows various organizational units of a web document to be transferred and browsed according to the

information contents. This will allow better utilization of the limited bandwidth. Similar effort has been reported in [66].

### **Mobile Data Security**

Security is a prime concern in mobile databases due to nature of the communication medium. New risks caused by mobility of users and portability of computers can compromise confidentiality, integrity and availability-including accountability. In a mobile database environment, it may be a good idea to have data summarized [67], so that only metadata can be stored on mobile platform and more detailed data can be kept only on the mobile service station (MSS). The higher frequency of disconnection also requires a more powerful recovery model. Such situations offer attackers the possibility of masquerade as either a mobile host or MSS. This needs a more robust authentication service [68]. Another issue is to maintain the privacy of location data of mobile hosts. Ideally, only mobile user and home agent should have knowledge about a mobile host's current position and location data. All user identification information, including message origin and destination, has to be protected. In order to achieve anonymous communication, aliases can be used or communication can be channelled through a trusted third party. Furthermore, the identity of users may also need to be kept secret from other MSSs. Access-based control policies can be adapted to provide data security on a mobile platform.

## **6. CONCLUSIONS**

Mobility brings in a new dimension to the existing solutions to the problems in distributed databases. We have surveyed some of the problems and existing solutions in that direction. We have highlighted the merits and demerits of existing solutions. We have identified some of the upcoming research areas that, due to the nature and constraints of mobile computing environment, require rethinking. The upcoming mobile database research directions discussed here will be the centres of attractions among mobile database researchers in years to come.

## **REFERENCES**

- [1] B. Bruegge and B. Bennington, "Applications of Mobile Computing and Communications," *IEEE Personal Communications*, vol.3, no. 1, Feb. 1996.

- [2] T. Imielinski and B. R. Badrinath, "Wireless Mobile Computing: Challenges in Data Management," *Communications of ACM*, 37(10), October 1994.
- [3] E. Pitoura and B. Bhargava, "Building Information Systems for Mobile Environments," *Proc. of 3rd International Conference on Information and Knowledge Management*, 1994, pp.371-378.
- [4] E. Pitoura and B. Bhargava, "Maintaining Consistency of Data in Mobile Computing Environments," *Proc. of 15th International Conference on Distributed Computing Systems*, June, 1995. Extended version appeared in *IEEE Transactions on Knowledge and Data Engineering*, 2000.
- [5] J. Ioannidis and G.Q. Maquire, "The Design and Implementation of a Mobile Networking Architecture," *USENIX winter 1993 Technical Conference*, Jan. 1993.
- [6] S. Mohan and R. Jain, "Two user location Strategies for PCS," *IEEE Personal Communications Magazine*, 1(1), 1<sup>st</sup> quarter, 1994.
- [7] D.J. Goodman, "Trends in Cellular and Cordless Communications," *IEEE Communication Magazine*, June 1991.
- [8] B. Awerbuch and D. Peleg, "Online Tracking of Mobile Users," *Journal of ACM*, vol. 42, no. 5, 1995, pp. 1021-1058.
- [9] B. Badrinath, R. Imielinski and A. Virmani, "Locating Strategies for Personal Communication Networks," *Proc. of IEEE GLOBECOM workshop on Networking for Personal Communications Applications*, Dec. 1992.
- [10] G. Cho and L.F. Marshall, "An efficient Location and Routing Scheme for Mobile Computing Environment," *IEEE Journal on selected Areas in Communications*, vol. 13, no. 5, June 1995.
- [11] N. Shivakumar and J. Widom, "User Profile Replication for Faster Location Lookup in Mobile Environments," *Proc. of the 1st ACM International Conference on Mobile Computing and Networking (Mobicom'95)*, Oct. 1995, pp. 161-169.
- [12] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, *Network Flows*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.
- [13] R. Jain, "Reducing Traffic Impacts of PCS using Hierarchical User Location Databases," *Proc. of the IEEE International Conference on Communicatios*, 1996.
- [14] M.V. Steen, F.J. Hauck, P. Homburg and A.S. Tanenbaum, "Locating Objects in Wide-area Systems," *IEEE Communication Magazine*, pp. 2-7, January 1998.
- [15] E. Pitoura and I. Fudos, "An Efficient Hierarchical Scheme for Locating Highly Mobile Users," *ACM proceedings for International Conference on Information and Knowledge Management (CIKM)*, 1998.
- [16] V. Anantharaman, M.L. Honig, U. Madhow and V.K. Wei, "Optimisation of a Database Hierarchy for Mobility Testing in a Personal Communication Network. Performance Evaluation," 20(13), May 1994.
- [17] P. Krishna, N.H. Vaidya and D.K. Pradhan, "Static and Dynamic Location Management in Mobile Wireless Networks," *Journal of Computer Communications* (special issue on Mobile Computing), 19 (4), March 1996.
- [18] S. Dolev and D.K. Pradhan, "Modified Tree Structure for Location Management in Mobile Environments," *Computer Communications*, vol. 19, 1997, pp.335-345.
- [19] J.S.M. Ho and F. Akyildiz, "Dynamic Hierarchical Database Architecture for Location Management in OCS Networks," *IEEE Transactions on Networking*, vol. 5, no. 5, Oct. 1997.
- [20] M. Satyanarayanan, "Mobile Information Access," *IEEE Personal Communications*, vol.3, No. 1, Feb. 1996.

- [21] M. Satyanarayanan, J.J. Kistler, B. Kumar, et al., "Coda: A Highly Available File System for a Distributed Workstation Environment," *IEEE Transaction on Computers*, vol. 39, no. 4, April 1990.
- [22] D. Barbara and T. Imielinski, "Sleepers and Workaholics: Caching Strategies in Mobile Environments," *VLDB Journal*, 1995.
- [23] K. Wu, P.S. Yu and M. Chen, "Energy Efficient caching for Wireless Mobile Computing," *Proc. of the 12<sup>th</sup> International Conference on Data Engineering*, New Orleans, Feb. 1996.
- [24] M.R. Ebling, "Evaluating and improving the Effectiveness of Caching for Availability," Ph.D. Thesis, Department of Computer Science, Carnegie Mellon University, 1997.
- [25] J. Cai, K.L. Tan and B.C. Ooi, "On Incremental Cache Coherency Schemes in Mobile Computing Environment," *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 1997.
- [26] O. Wolfson and S. Jajodia, "Distributed Algorithms for Dynamic Replication of Data," *Proc. of the Symposium on Principles of Database Systems*, CA, 1992, pp. 149-163.
- [27] B.R. Badrinath, and T. Imielinski, "Replication and Mobility," *Proc. of 2nd IEEE Workshop on the Management of Replicated Data*, Nov. 1992, pp. 9-12.
- [28] K. Ravindran and K. Shah, "Casual Broadcasting and Consistency of Distributed Data," *Proc. of 14th International Conference on Distributed Computing Systems*, June 1994, pp. 40-47.
- [29] M. Faiz and A. Zaslavsky, "Database Replica Management Strategies in Multidatabase Systems with Mobile Hosts," *Proc. of 6th International Hong Kong Computer Society Database Workshop*, 1995.
- [30] Y. Huang, P. Sistla and O. Wolfson, "Data Replication for Mobile Computers," *Proc. of the ACM SIGMOD International Conference on Management of Data*, 1994.
- [31] J. Gray, P. Helland, P. O'Neil and D. Shasha, "The Dangers of Replication and a Solution," *Proc. of ACM SIGMOD International Conference on Management of Data*, 1996, pp. 173-182.
- [32] S. Wu and Y. Change, "An Active Replication Scheme for Mobile Data Management," *IEEE Proceedings of 6<sup>th</sup> DASFAA*, Taiwan, 1999.
- [33] N. Barghouti and G. Kaiser, "Concurrency Control in Advanced Database Applications," *ACM Computing Surveys*, vol. 23, no. 3, 1991, pp.269-317.
- [34] K. Ramamritham and P.K. Chrysanthis, "A Taxonomy of Correctness Criterion in Database Applications," *Journal of Very Large Databases*, vol. 4, no. 1, Jan. 1996.
- [35] G.D. Walborn and P.K. Chrysanthis, "Supporting Semantics-Based Transaction Processing in Mobile Database Applications," *Proc. of 14th IEEE Symposium on Reliable Distributed Systems*, Sept. 1995, pp.31-40.
- [36] J. Kisler and M. Satyanarayanan, "Disconnected Operation in the Coda File System," *ACM Transactions on Computer Systems*, vol. 10, no. 1, 1992.
- [37] S.K. Madria and B. Bhargava, "Improving Availability in Mobile Computing Using Prewrite Operations," *Distributed and Parallel Database Journal*, Sept. 2001.
- [38] Q. Lu and M. Satyanaraynan, "Improving Data Consistency in Mobile Computing Using Isolation-Only Transactions," *Proc. of the fifth workshop on Hot Topics in Operating Systems*, Orcas Island, Washington, May 1995.
- [39] P.K. Chrysanthis, "Transaction Processing in a Mobile Computing Environment," *Proc. of IEEE workshop on Advances in Parallel and Distributed Systems*, Oct. 1993, pp.77-82.
- [40] S.K. Madria and B. Bhargava, "A Transaction Model for Mobile Computing," *Proc. 2<sup>nd</sup> IEEE International Database and Engineering Application Symposium (IDEAS'98)*, Cardiff, U.K., 1998.

- [41] S.K. Madria and B. Bhargava, "On the Correctness of a Transaction Model for Mobile Computing," *9<sup>th</sup> Intl. Conf. on Database and Expert System Applications (DEXA '98)*, Vienna, Austria, Aug. 1998, *Lecture Notes in Computer Science*, vol. 1460, Springer-Varleg.
- [42] S.K. Madria and B. Bhargava, "System Defined Prewrites to Increase Concurrency in Databases," *Proc. of First East-European Symposium on Advances in Databases and Information Systems* (in co-operation with ACM-SIGMOD), St.-Petersburg, Sept. 1997.
- [43] M. H. Eich, and A. Helal, "A Mobile Transaction Model That Captures Both Data and Movement Behavior," *ACM/Baltzer Journal on Special Topics on Mobile Networks and Applications*, 1997.
- [44] C. Pu, G. Kaiser and Hutchinson, "Split-transactions for Open-ended Activities," *Proc. of the 14th VLDB Conference*, 1988.
- [45] L.H. Yeo, and A. Zaslavsky, "Submission of Transactions from Mobile Workstations in a Co-operative Multidatabase Processing Environment," *Proc. of the 14th IEEE International Conference on Distributed Computing Systems (ICDCS'94)*, June 1994.
- [46] R.A. Dirckze and L. Gruenwald, "A Toggle Transaction Management Technique for Mobile Multidatabases," *ACM Proc. of International Conference on Information and Knowledge Management (CIKM)*, 1998.
- [47] D.B. Terry, D. Goldberg, D.A. Nichols and B.M. Oki, "Continuous Queries over Append-only Databases," *Proc. of the ACM-SIGMOD International Conference on Management of Data*, June 1992.
- [48] S. Acharya, R. Alonso, M. Franklin and S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication environments," *Proc. of the ACM SIGMOD Conference*, California, 1995.
- [49] Vitria Technology Inc. , <http://www.vitira.com>.
- [50] E. Pitoura and P.K. Chrysanthis, "Scalable Processing of Read-only Transactions in Broadcast Push," *Proc. of IEEE International Conference on Distributed Computing Systems*, 1999.
- [51] E. Pitoura and P.K. Chrysanthis, "Exploiting Versions for Handling Updates in Broadcast Disks," *Proc. of VLDB*, 1999.
- [52] J. Shanmugasundaram, A. Nithrakashyap, R. Sivasankaran and K. Ramamritham, "Efficient Concurrency Control for Broadcast Environments," *Proc. of the ACM SIGMOD Conference*, 1999.
- [53] D. Barbara, "Certification Reports: Supporting Transactions in Wireless Systems," *Proc. of 17<sup>th</sup> International Conference on distributed Computing systems*, 1997, pp. 466-473.
- [54] S. Lee, C. Hwang and H. Yu, "Supporting Transactional Cache Consistency in Mobile Database Systems," *Proc. of ACM International workshop on Data engineering for Wireless and Mobile Data Access*, 1999.
- [55] S.K. Madria, B. Bhargava, E. Pitoura and V. Kumar, "Data Organization Issues in Location Dependent Query Processing in Mobile Computing Environment," *Proc. of 4<sup>th</sup> East-European Symposium on Advances in Databases and Information Systems* (in co-operation with ACM-SIGMOD), Prague, Czech Republic, 2000.
- [56] T. Imielinski and B.R. Badrinath, "Querying in Highly Distributed Environments," *Proc. of the 18<sup>th</sup> VLDB Conference*, 1992, pp. 41-52.
- [57] O. Wolfson, X. Bo, C. Sam and L. Jiang, "Moving Objects Databases: Issues and Solutions," *Proc. of SSDBM*, 1998, pp. 111-122.



- [58] O. Wolfson, A.P. Sistla, C. Sam and Y. Yelena, "Updating and Querying Databases that Track Mobile Units," *Distributed and Parallel Databases*, vol. 7, no. 3, 1999, pp.257-387.
- [59] G. Dong and M. Mohania, "Algorithms for View Maintenance in Mobile Databases," in *proceedings of the First Australian Workshop on Mobile Computing and Databases and Applications (MCDA '96)*, Melbourne Australia, 1996.
- [60] Y. Huang, P. Sistla and O. Wolfson, "Divergence Caching in Client-Sever Architectures," *Proc. of the third International Conference on Parallel and Distributed Systems (PDIS)*, Austin, TX, Sept. 1994, pp. 131-139.
- [61] P. Sistla and O. Wolfson, "Temporal Conditions and Integrity Constraints in Active Database Systems," *Proc. of the ACM SIGMOD International Conference on Management of Data*, May 1995, pp. 269-280.
- [62] I. Stanoi, D. Agarwal, A. El Abbadi, S.H. Phatak and B.R. Badrinath, "Data Warehousing Alternatives for Mobile Environments," *Proc. of ACM International Workshop on Data Engineering for Wireless and Mobile Access*, Seattle, Washington, 1999.
- [63] G. Alonso, R. Gunthor, M. Kamath, D. Agrawal, A. El Abbadi and C. Mohan, "Exotica/FMDC: Handling Disconnected Clients in a Workflow Management Systems," *Proc. of 3rd International Conference on Co-operative Information Systems*, May 1995.
- [64] G.M. Voelker and B.N. Bershad, "Mobisaic: An Information Management System for Mobile Wireless Computing Environment," T. Imienlinski and H. Korth, editors, *Mobile Computing*, Kluwer Academic Publishers, 1996, pp. 375-395.
- [65] M. T. Stanley, Y. H. Va Leong, D. McLeod and A. Si, "On Multi-Resolution Document Transmission in Mobile Web," *SIGMOD Record*, vol. 28, no. 3, 1999, pp.37-42.
- [66] M. Gaedke, M. Beigl, H. Gellersen and C. Segor, "Web Content Delivery to Heterogeneous Mobile Platforms," *ER workshops Proc. as Lecture Notes in Computer Science*, vol. 1552, Springer, 1998.
- [67] S.K. Madria, M. Mohania and J. Roddick, "A Query Processing Model for Mobile Computing using Concept Hierarchies and Summary Databases," *Proc. of 5<sup>th</sup> Intl. Conference on Foundation for Data Organization (FODO'98)*, Japan, Nov. 1998.
- [68] B. Bhargava, S. Kamisetty and S.K. Madria, "Fault Tolerant Authentication in Mobile Computing," *Sp. Session, New Paradigms in Computer Security, at International Conference on Internet Computing, (IC'2000)*, Las Vegas, USA, pp. 395-402.