

Optimistic Caching Mechanism to Support Data Availability in Disconnected Mobile Environment

¹Sallam O Fageeri, ²Rohiza Ahmad, ³A.H. Muhamad Amin

Department of Computer and Information Sciences, Universiti Teknologi PETRONAS Tronoh, Malaysia

Abstract: Mobile technology is growing rapidly day by day. Now we are a witness of a huge plethora in using mobile devices for conducting business transactions that involve data from a remote database. But there are many issues that arise when using mobile clients in business such as network connectivity, power consumption and low bandwidth. In this study we consider the client side data caching to increase the content of a particular mobile device cache itself as during the period of disconnection the device will have to depend on its own resources rather than the server or some other peers. This can be done by ensuring only the most relevant data will be cached and the data should be kept in the cache in a new structure that can boost the amount of data kept. In doing so, greatly improve performance in terms of cache hit rate, average query latency and query generate duration.

Keywords: Data management, data caching, data replication, mobile computing

INTRODUCTION

Mobile technology has gained increasing attention from the community due to its ability to provide services at any time and place. To date, many applications have been developed for the mobile platform. Among them are query for billing, weather forecasting as well as retrieval of information from remote servers. (Tarafdar and Haghjoo 2010).

However, before seamless operation in the mobile environment can be guaranteed, many challenges will need to be overcome first. For mobile computing, these challenges directly emanate from two main sources: the limited capabilities of mobile devices and the limited capabilities of mobile networks as compared to desktop computing devices and wired networks, respectively. Limited capabilities of mobile hardware results from the inherent limitations of the mobile environment and the small size of mobile devices; both of them cannot be completely overcome. Each of these hardware limitations presents a different set of challenges that need to be addressed not only by physically upgrading or enhancing the hardware but also in providing software that can enhance the use of the existing hardware. Mobile environments lead to new operating paradigms that have not been encountered previously in the traditional distributed computing environments. Thus, this new paradigm creates another unique set of challenges that need to be addressed in the software development of mobile applications.

Data retrieval is one popular component usually supported by mobile applications. For example, weather

forecasting involves the retrieval of data or information from a certain website. Similarly for query billing, in order for the requested server to send the billing information to the client device, the information will need to be fetched first from the internal database. Hence, where the data is stored as well as how the data is structured will make a difference to the performance of the data retrieval process. Thus, data management is an important aspect that needs to be carefully planned to support performance. Of course this is not only true for the mobile environment but also to the other environments, i.e., wired, wireless etc. However, for the mobile environment, the challenges of data management are even more. Not only performance is a concern, data availability is another issue that needs to be addressed.

Since it is a common knowledge that the mobile environment is prone to disconnections which can be voluntary or involuntary (Vora *et al.* 2005), some measures are needed in order to ensure data retrieval transactions will still be successful even during these periods of disconnection. Hence, in this study, a method based on prefetching technique is proposed to handle the data availability issue. The method will copy most relevant data into the client cache when the client is about to be disconnected from the network. Besides deciding what data to cache, the method will also restructure the data for maximum amount of cache. With such a method, the number of cache miss is anticipated to be able to be reduced.

The next two sections present a mobile computing environment architecture and its challenges. They are

followed by a section on related work where in particular some of the existing caching methods which could be adopted and enhanced in our work is discussed. This will be continued with a section on the proposed solution and a section on conclusion.

MOBILE ARCHITECTURE

In a mobile computing environment, a typical architecture as shown in Fig.1 will consist of mobile units (MUs), base stations (BSs) or mobile support stations (MSS) and fixed hosts (FHs). MUs are battery powered portable computers which can be in the form of mobile phones, PDAs, notebooks or any other devices that have mobile access. MUs are connected directly to BS via wireless channels. MUs can move around freely in a restricted area, which is referred to as the ‘‘geographical region’’ (G). For example in Fig.1, G is the total area covered by all BSs. This cell size restriction is mainly due to the limited bandwidth of the wireless communication channels. To support the mobility of MUs and to exploit frequency reuse, the entire G is divided into smaller areas called cells. A particular BS manages each cell. Each BS will store information such as user profile, login files and access rights together with user’s private files. At any given instant, an MU communicates only with the BS responsible for its cell. The mobile discipline requires that MU must have unrestricted movement within G (inter-cell movement) and must be able to access desired data from any cell under the same BS management (Kumar & Mohania 2002). An MU changes its location and network connections while computations are being processed. While in motion, an MU retains its network connections through the support of BSs with wireless connections.

Fixed hosts and Base stations perform the transaction and data management functions with the help of 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 both. Within this mobile computing environment, shared data are stored and controlled by a number of DBSs. BSs will provide commonly used application software so that a mobile user can download the software from the closest FH and run it on the MU or execute it remotely on the FH. The decision to install and execute an application locally or request the FH to do the execution largely depends on the capability of the MUs themselves. Some MUs, for example notebooks, may have some server capability to perform computations locally using local concurrency control and recovery algorithm, while others such as the older version mobile phones, may have very slow CPU and very little memory and thus, acts as an I/O device only. They depend on some FHs to perform computations for them.

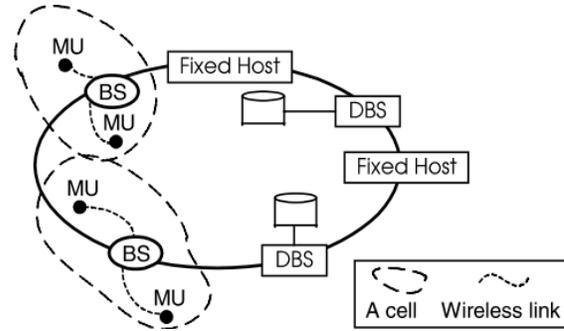


Fig. 1: Mobile computing architecture (Kumar and Mohania 2002)

As stated earlier, an MU can go anywhere within G without the fear of disconnection. However, it is also allowed to leave G, but this requires extra processes. When an 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 an MU and is responsible for maintaining end-to-end data movement connectivity (Kumar and Mohania 2002). Thus, during this period of disconnection, data availability will be an issue.

Mobile technology challenges: The limitations of the mobile devices or MUs and the network, along with the introduction of the mobile operating paradigm, introduce new challenges for the research community to solve. For example, a simple inventory system which works fine on a desktop cannot be guaranteed to behave the same on the mobile devices. Hence, new approaches will need to be developed in order to handle the requirements of the mobile devices and the mobile network. The next sub-sections list the limitations brought along by mobile devices and the mobile network as well as some of the existing solutions or works done in handling them.

Limited resources on mobile devices: While the resources on mobile devices, such as processing power, storage and memory, are increasing stupendously, they are however, still much lower than their desktop counterparts. The small form-factor of mobile devices, especially PDAs and mobile phones, limits the space that can be used for adding more powerful components. Hence, size is the biggest bottleneck in increasing the capabilities of mobile devices. Therefore, the resources on mobile devices will always be limited compared to desktop computers. Yet, if the resources are consumed conservatively, they are adequate for supporting sophisticated applications. Therefore, methods to operate in Mobile environments need to be developed

so that they are sensitive to the miniscule capabilities of mobile devices, by using resources on it more efficiently.

Limited battery power: Mobile devices are powered by finite capacity rechargeable batteries. Without continuous power supply, their life is limited (in the order of a few hours for laptop computers and a bit longer for smart phones and PDAs), whereas desktop computers are continuously powered and have virtually infinite power. In fact, limited battery life is an important reason for mobile devices being fitted with resources that are not very powerful but consume less energy. It is, therefore, very important to develop methods that utilize battery efficiently.

Low bandwidth and high latency networks: Mobile networks typically have much lower bandwidth and higher latency compared with wired networks; which on the outset makes them unsuitable for supporting data intensive applications. The difference in bandwidth can be several orders of magnitude lower for wireless networks (for example, GPRS based networks can support up to a few hundred Kbps compared with several Mbps residential broadband connections). The limited spectrum availability severely limits the ability of mobile networks to scale concurrently in providing higher bandwidth as well as supporting additional users. It is, therefore, prudent for methods designed to operate in mobile environments to use network bandwidth efficiently.

Limited coverage of mobile networks: While traditional cellular voice networks provide near ubiquitous coverage, high speed data connections are currently limited to areas with high user density. Besides, obstacles such as buildings and mountains along with signal loss and various types of noise do not allow radio signals to reach every conceivable part of the coverage area. This will always lead to some mobile users without network connectivity for various durations. Therefore, methods used for operating in mobile environments cannot assume continuous network coverage. Strategies to overcome temporary disconnections need to be considered.

Disconnections: The above discussed challenges together create an inevitable mode of operation for mobile users: Disconnected. Technological and natural limitations restrict the ability of mobile users to always remain connected to mobile networks. This results in involuntary disconnections that cannot be overcome. In many cases, users are obliged to pay for the duration of network connections. Besides, intangible costs increase in the presence of wireless network connections. Under such circumstances, users could deliberately disconnect to conserve resources. This results in voluntary

disconnections. Such disconnected modes of operation are not common in wired environments. Hence, new methods that can operate in the presence of disconnections need to be developed for mobile environments. Many challenges discussed above are the results of physical or hardware limitations. Some limitations such as limited resources on mobile devices and inadequate spectrum cannot be overcome entirely; whereas, the rate of improvement of other limitations such as limited battery life is painfully slow (Starner 2003). Therefore, these challenges need to be addressed by the software that is running the wireless infrastructure, the mobile devices and mobile applications. Infrastructure software should anticipate connectivity problems and undertake measures to deal with them, while supporting mobile users that do not get good connectivity. Mobile device software needs to be aware of the implications of its dwindling resources for every action it initiates. Mobile applications need to minimize the strain on the computing resources of mobile devices by reducing processing and using efficient data transfer protocols.

Mobile devices such as mobile phones and laptops, is now used everywhere, where they help to use internet applications of the simple databases and there is a clear difference between dealing with the rules of wireless data and databases traditional and the ratio of the interruption of repeated because of mobility and bandwidth that is usually influenced by low It is known also there is a consumption increased resources when the query is issued from the client to the server, which led to seek to reduce the number of requests issued to the server, where the opposite happens when the issuance of the request from the server to the client, knowing that he cannot rely on the client's storage technologies timer because it is not efficient with regard to energy and interruption (Rathore 2008).

RELATED WORKS

Mobile Data Caching: As being defined by Wang *et al.* (2008), caching is considered as an important technique that can be used to copy data from remote location to the local memory of mobile devices. The main purpose of caching is basically to reduce access to the server which is known to be more costly as compared to access to local storage. On top of that, caching can also reduce query response, save bandwidth and improve system performance (Tian & Denko 2007). Owing to the disconnection and the mobility of the mobile clients, classical cache management strategies may be inappropriate for mobile environments. Generally, cache placement, cache discovery, cache consistency and cache replacement techniques constitute cache management in the mobile environment (Anandharaj and Anitha 2009).

Caching usually means storing frequently accessed data objects in the local buffer of an accessing device

(Vora *et al.* 2005). Caching has also been used for reducing delays and improving performance in mobile environments. An additional advantage of caching in mobile environments is that the cached data can be accessed without the presence of network connection. Simple cache scheme (Joonho Cho *et al.* 2003) (H. Artail *et al.* 2005). For example, using caching techniques to enhance data access performance in terms of the time taken to fulfill a request. Whenever data are requested by clients, the local cache will be checked first and if the requested data is not found then the request will be sent to the server for fulfillment. Simple Cache works well as long as the connection to the data server is reliable and not too expensive; otherwise, it results in failed data requests or request timeouts (Yin, L *et al.* 2006) (Das 2004). However, in the mobile environment connection cannot be guaranteed all the time. Hence, to overcome the problem of data unavailability, more data should be cached at the client to improve hit ratio. But, this cannot be easily done if the client has a small storage space such as the mobile devices. Therefore, approaches such as cooperative and collaborative caching (Du, Y., and Gupta 2005) (Narottam Chand 2006) have been known to be used as cache placement's strategy for mobile accessing devices in the wireless sensor networks and ad hoc networks. In cooperative and collaborative caching, data to be cached are placed or rather distributed among several nodes in the network (Du, Y., and Gupta 2005). Hence, what sort of data can be kept in ones local cache can be in the form of data (i.e., cache-data), path (i.e., cache-path) or both (i.e., hybrid-cache) (Yin, L *et al.* 2006). In cache-data, frequently accessed data from the main server passes through a network of nodes. Each node will contain some portion of the cache data. Later, if one of the mobile devices in the network is requesting for some data, the node closest to it will either provide the data if it has them or redirect the request to the other node along the path if it hasn't. This will go on until the request has been fulfilled or the main server has been reached. Cache-path as in Fig. 2 saves space by storing locations where the data should be stored, while cache data saves time by storing the data instead of the path. The third scheme, hybrid-cache, is a middle solution where queries are cache by path or by data, depending on what is optimal (Yin, L *et al.* 2006).

Even though cooperative and collaborative caching approaches can maximize the amount of cached data, they have a problem in terms of performance in cache location discovery. Thus, many approaches have been proposed to handle this problem. One of the approaches is called a Zone Cooperative (CZ) (Narottam Chand 2006). In CZ, the set of one-hop neighbors in the transmission range forms a zone. The mobile neighbor (MN) in the forms of cooperative cache, the origin of each data item is held by a particular data source act as a server. Data is updated only at the server. Each MN store the frequently accessed data items in its cache.

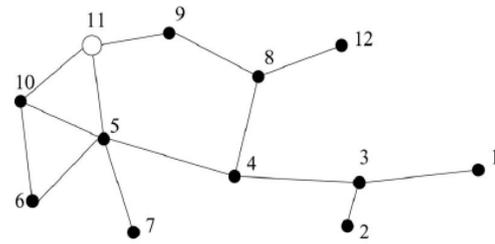


Fig.2 Caching path in MANET (Yin, *Let al.* 2006)

The data items in the cache satisfy not only the node's own requests but also the data requests passing through it from other MN. Once the data is updated, MN cached copy becomes invalid, After Zone cache miss, the request is forwarded to the neighbor along the routing path. Each MN searches the data in its local cache or zone cache before forwarding the request to the next node that lies on a path towards data server. Another almost similar approach is called Group Caching (GC) (Narottam Chand 2006). In GC each node and its one-hop neighbors form a group within transmission range by sending the "Hello" messages periodically. Each group has a master node which needs to communicate with its members directly through one-hop routing. It increases the communication speed and reduces the delay among nodes in the group. In order to utilize the cache space on each node in a group, the master node checks the caching status of group members using caching control message. Hence, it stores more different data items and increases the data accessibility. In order to record the caching status of group members, each node maintains self-table, group table to record caching status of itself and its group members respectively. When a MN receives the request, it checks the self-table for local cache hit. After local cache miss, checks the group table to send the request to group members for a remote cache hit. After the remote cache hit, the request send to the next group along the path to the server.

Neighbor-Caching (NC) (Cho, J., S. Oh, *et al* 2003) on the other hand, utilizes the cache space of inactive neighbors for caching tasks. The basic operations of NC are as follows. When a node fetches a data from a remote node, it puts the data in its own caching space for reuse. This operation needs to evict the least valuable data from the cache based on a replacement algorithm. With this scheme, the data that is to be evicted is stored in the idle neighbor node's storage. In the near future, if the node needs the data again, it will request the data not from the far remote source node but from the nearest neighbor that keeps the copy of the data. The NC scheme utilizes the available cache space of neighbor to improve the caching performance. However, it lacks of the efficiently cooperative caching protocol among the MHs.

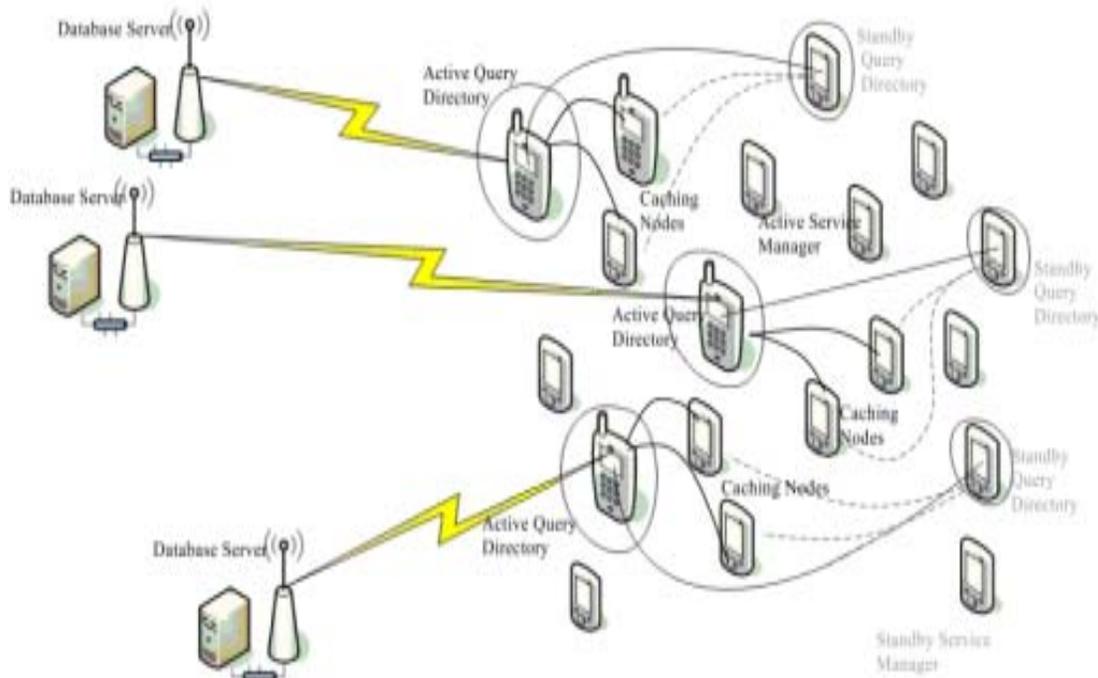


Fig.3 DB query caching (H. Artail *et al.* 2005)

Besides caching of data or path, caching of database queries and their responses has also been suggested as a way to enhance retrieval performance. As shown in Fig. 3, the queries cached were used as indexes to the responses when searching for data. The described architecture has an elected service manager (SM) that oversees the caching operations and is responsible for assigning the roles of Query Directory (QDs) and Caching Nodes (CNs) to specific mobile nodes for caching queries and their responses, respectively. The QDs decide on which CNs to cache external (non-cached) data and maintain one-way hash tables to quickly locate the responses of the locally cached queries. The limitations of this scheme include relying heavily on the SM (and its backup), which, just like any other mobile node, may move around, leave the network, or run low on battery power(H. Artail *et al.* 2005).

An approach known as COOP (Du, Y., and Gupta 2005) is another alternative solution to cache discovery. COOP as shown in Fig. 4 works by maintaining a detailed history (source and position of every “passing” data) in the nodes of a MANET and defining cooperation zones. Each node maintains a table where previously received requests are recorded. Initially, a node checks its table after its local misses and before flooding a request. If matching entries are found, the node compares its distances to these matching caches and the original data source and selects the closest node to redirect the request. In case

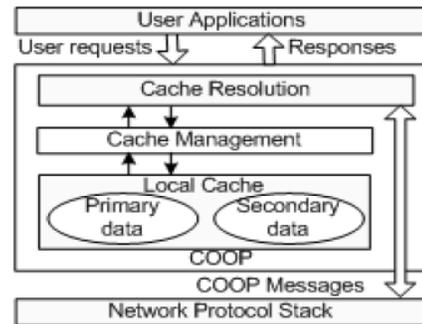


Fig.4 COOP caching (Du, Y., and Gupta 2005)

of table miss, the requester node is using an adaptive z-hop flooding technique. If adaptive flooding also fails, the request is sent to the data center.

Besides looking at the other nodes one at a time for requesting data, a shortcut approach has been introduced in (S. Lim, W. Lee, G. Cao 2006). This approach as being presented uses the technique of broadcasting a data request to the entire network. All nodes that have the requested data are supposed to send an acknowledgment back to the source of the broadcast. The Requesting Node (RN) will then issue a request for the data (unicast) to the first acknowledging node it hears from. However, the scheme is inefficient in terms of bandwidth usage because of the broadcasts, which, if occur frequently, will largely decrease the throughput of the system due to flooding the network.

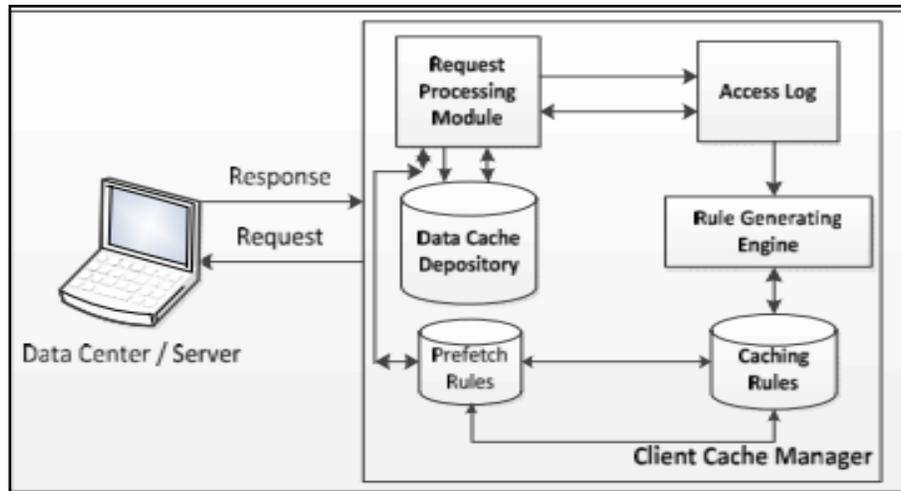


Fig 5 Pre-fetching in MANETs (Chauhan *et al.*, 2011)

Basically, in order to ensure minimal cache miss, the data cached must be relevant in the first place. Hence, for this, a pre-fetching technique as shown in Fig. 5 has been suggested in (Chauhan *et al.* 2011). With pre-fetching as an integrated part of data caching, the technique will try to sense the future needs of Mobile Nodes (MNs), once it is established regarding the requirement of a particular data item, data is fetched and kept in the cache before it is required, but pre-fetching consumes a large amount of system resources such as computation power and energy. A heuristic algorithm is proposed to decide whether a caching node shall pre-fetch a data item or not. The basic idea is that the pre-fetching cost should be compensated by future minimization of access delay and energy cost. To conduct this algorithm, the network topology must be static and the data access probability must be already known for all the nodes in the network.

Due to the possibility of the original data on the server to be changed, in the traditional wired environment, old caching techniques require the server to be in regular contact with the client for maintaining client cache (He 2006). However, in the mobile environment where connectivity cannot be guaranteed, several approaches have been proposed for cache replacement such as the approach presented in (Barbara and Imielifiski 1995). In the approach, the server periodically broadcasts an Invalidation Report (IR) in which the changed data items are indicated. Rather than querying the server directly regarding the validation of cached copies, the clients can listen to these IRs over the wireless channel and use them to validate their local cache. The IR-based solution is attractive because it can scale to any number of clients who listen to the IR. However, the IR-based solution has some major drawbacks such as long query latency and low bandwidth utilization. Besides the above, another approach for data replacement was addressed in (Cao

2003). In the approach, a small fraction of the essential information (called Updated Invalidation Report (UIR)) related to cache invalidation is replicated several times within an IR interval. Using this approach, the client can answer a query without waiting until the next IR. However, if there is a cache miss, the client still needs to wait for the data to be delivered. Other than the above, in (Myers 1997) a new hybrid adaptive caching technique, which combines page and object caching to reduce the miss rate in client caches dramatically, is presented. (HAC) is a novel technique for managing the client cache in a distributed, persistent object storage system, while in (Vakali 2001), an overview of a series of web cache replacement algorithms based on the idea of preserving a history record for cached Web objects is presented. The number of references to Web objects over a certain time period is a critical parameter for the cache content replacement. In (ARI *et al.* 2002), the preliminary design of an adaptive caching scheme using multiple experts, called ACME is described. ACME is used to manage the replacement policies within distributed caches to further improve the hit rates over static caching techniques.

Mobile Data Replication: Data replication is another known approach to handle data unavailability problem. Data replication stores multiple copies of data on different servers distributed over networks. Generally the aims of such action are to further improve data retrieval performance as well as maximize productivity and availability of data required by users. As for the application of data replication in the mobile environment, further developments of the method are still needed (Monteiro 2007) (Bouwman *et al.* 2009) (Zaslavsky *et al.* 1996) (Ding *et al.* 2001) (Khairullah 2009).

Applied algorithm called (ERDA) which is based on Remote Data Access (MS RDA) the address and the

organization to bridge the gap existing in the algorithm and the reform of the structure of databases and replication in the cell so as to avoid overlapping groups of data on the relationship between the server and the client through to find a way appropriate division of these relations, an example of this geographical location and through which can reduce the number of replicas and avoid conflicts (Itani *et al.* 2005).

A Meta dynamic update protocol in replicated database, allows adjustment scenarios in which they can change the consistency and coherence, according to the requirements of the application needs. And showed the extent of basic services for distributed systems and reliable messaging and opinions can help to get this kind of functionality. He claimed that can be integrated into such a protocol definition and the commercial database programming interfaces by means of the orientation and therefore, is currently a prototype implementation of this Protocol, the current definition. It is important to achieve deeper relationships between the views and problems update protocol, as it had done with the consensus and agreement problems other features offered are an initial step in the definition of the update problem for systems of transactions replicated, further research that show and can be done by the "isolation" between the copies and control protocols , protocol executions metadata can be defined as characteristics of the strongest and weakest on the basis of "dynamic update protocol levels of isolation," (Fritzke Jr, U., R. P. Valentim, *et al.* 2008).

A new replication strategy known as (Cedar) works on preserving eventual consistency by using a Client/Server design; a central server holds the master copy of the database. At infrequent intervals when a client has connected to the server (which may occur hours or days apart), its replica is refreshed from the master copy. The central server is used to hold the master copy in this strategy (Tolia *et al.* 2007).

Walter is new techniques for geo-replica which had properties as storage system for web applications, Walter work on a techniques to avoid conflict occurs in across sites , thereby allowing to implement commit locally in sites, a precisely-stated isolation property that permits asynchronous replication across sites without the need for conflict resolution , A key feature behind Walter is Parallel Snapshot Isolation (PSI), PSI thus permits an efficient, while also providing strong guarantees to applications demonstrated the usefulness of Walter by building a Facebook like application of social networking and porting a third-party Twitter clone , applications were simple to implement and achieved reasonable performance (Y.Sovran *et al.* 2011).

A new strategy based on a hybrid approach , built to exploit the features of optimistic and pessimistic replication, a strategy built on dividing data into frequently changed data and infrequently changed data,

update can allow or facing restricted according to this type, the propagation protocol achieves load balance in propagation and ordering process because, process operation shared by various hosts (Ahmed *et al.* 2011).

New replica propose which is built in the idea of using method for recording transaction data for all work done in database source, and forwards the change to the target , which use the method of shadow table and trigger , there is disadvantages of increasing space requirement because there is periodical detection of the table , which is increasing the process time (Khairullah 2009).

Replication strategy, that's has different ways of replicating and managing data on fixed and mobile networks. In the fixed network, the data object is replicated synchronously to all sites in a manner of logical three dimensional grid structure, while in the mobile network, the data object is replicated asynchronously at only one site based on the most frequently visited sites. In this strategy, the synchronous replication hinders the fixed network to be scaled in wide areas The proposed replication technique is compared with a baseline replication technique and shown to exhibit high availability, fault tolerance and minimal access times of the data and services, which are very important in an environment with low-quality communication links (J.Abawajy 2006).

Transaction-Level-Result-Set Propagation is proposed for mobile database replication, each fixed and mobile nodes store a copy replica of data. The mobile node is allowed to update its local replica. However, updates locally committed at the mobile nodes need to be checked at the fixed node before they can be globally committed (Ding *et al.* 2002).

Currently from literature review, the main approaches used to handle data availability are data replication and data caching.

Previous work in mobile data caching concentrate on how to increase the availability of data by using cooperative, collaborative and semantic caching, but if we trace all this work, mobile unit will not be able to proceed working in partial or totally disconnected, because some parts of cached data stored in other places like neighbor nodes, this concept will not support availability because mobile unit will depend on other location to answer any query, also, a node which having a cached data may not be reachable (Joonho Cho *et al.* 2003) (Yin, L *et al.* 2006) (Das 2004). In semantic approach query will be answered from the previous cached queries answers, in this case if different query issued cache missed may occur (Mershad & Hassan Artail 2009).

PROBLEM FORMULATION

During the literature review, existing works which focus on data availability have been reviewed. From the

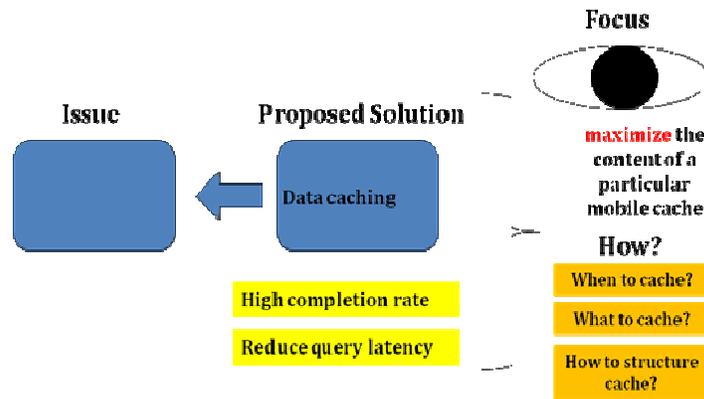


Fig.6 Proposed solution for Mobile data caching

review exercise, unavailability of data has been identified as a problem in the mobile environment and caching and data replication have been suggested as the means for solving this problem. However, from the literatures that have been reviewed the approaches suggested by them were all very much dependent on the accessing device to be connected either to the server or to the peers in its collaborative network. Hence, none has focused on data availability issue in a standalone disconnected mobile device. The reason for this could be due to the small storage capacity of the device that makes it difficult to store valuable data that can be used during the period of disconnection. However, this has become a research challenge which if overcome can have a great benefit to the mobile environment area.

Hence, in this study, the issue of data unavailability during the period of disconnection in the mobile environment has been identified as a problem to be solved. As during the period of disconnection, the mobile accessing device is pretty much acts as a standalone machine, data caching technique which can ensure maximum data availability becomes the aim of the study. Hence, when and what data to be sent to the cache as well as in what structure should the data be organized in the cache will be explored in this study. We believe that by maximizing relevant data availability, query misses can be reduced and performance in terms of time and other costs can be minimized.

Proposed solution: To begin the discussion of the proposed method, Fig.6 is presented to show the overall idea of the work.

In this study, the issue of maximizing the data availability during the period of disconnections is of concern. As mentioned in the problem formulation caching will be used as the technique for solving the said problem. However, rather than trying to develop a method that involves a lot of caches for maximizing

data availability like in the MANET environment, the proposed approach will be looking at ways to increase the content of a particular mobile device cache itself as during the period of disconnection the device will have to depend on its own resources rather than the server or some other peers (Fageeri et al. 2012). This can be done by ensuring only the most relevant data will be cached and the data should be kept in the cache in a new structure that can boost the amount of data kept. In doing so, performance of queries in terms of completion rate and latency shall be considered.

Proposed system architecture: Figure 7 shows where the proposed solution might be applied in the usual mobile environment architecture. We anticipate the period of disconnection could be involuntary as depicted by the critical area in Fig. 7 or voluntary inside the shown boundaries in similar figure. Involuntary disconnection could be due to the network and are a result of the unexpected loss of communication that is caused by external factors and voluntary to reduce communication cost or conserve battery power.

Proposed system framework: The system framework consists of mobile client side and application server side. The application server side components are used cache manager which is responsible of caching local data and working as a medium between the application server and the mobile client, cache content supports high speed access and a database management system. On the other side the mobile client contains an automatic network signal detector, a cache manager and lightweight database to maximize the cache content as well as accommodating and perform database actions.

Mobile client data caching scenario:

- As it is shown in Fig. 8, mobile clients use a mechanism to detect the poorness of network signal and then issue alert to mobile server (MS).

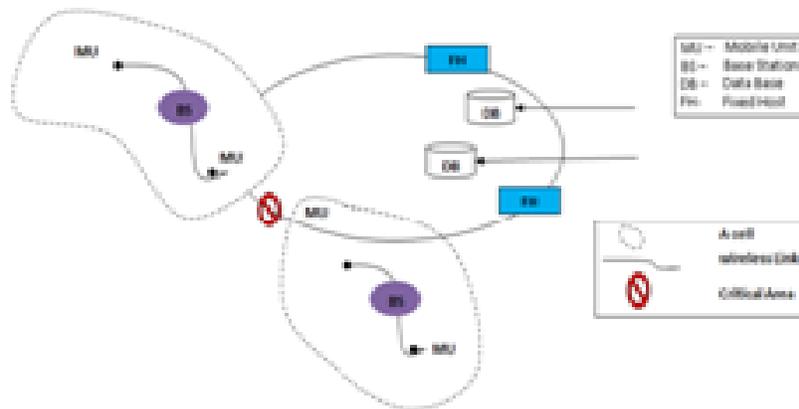


Fig.7: Solution placement in the Mobile architecture

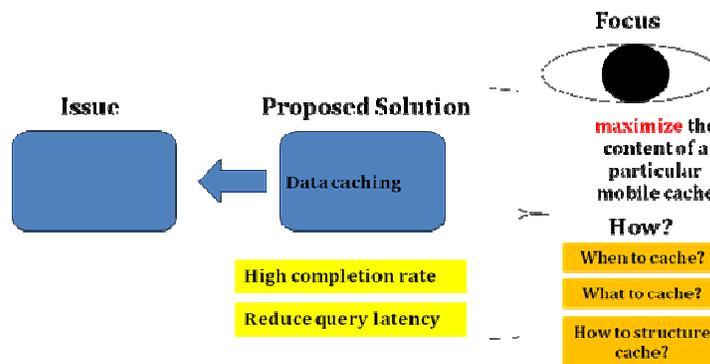


Fig. 8 General Diagram for mobile data caching

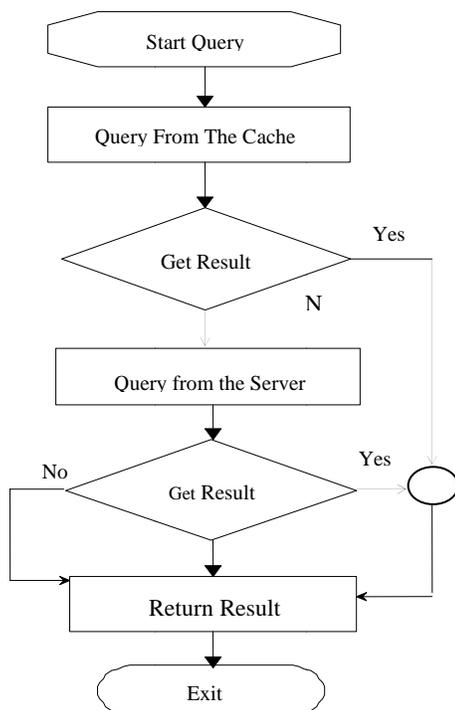


Fig. 9: Mobile client query model

MS will check the mobile unit (MU) information and the memory size then send a ready cached data to MU. The cache manager in MS will organize how to cache the requested data to local cache in MU.

- If the mobile client sends another alert, the server will check if it's the first time for the request. If so, then cached data will be sent without any notification. If not, the cached data in mobile client will be checked and if an update is necessary an updated data will be sent to MU along with the notification.
- In the structural design of the cached data in MU, we will use a lightweight database as shown in Fig.8, in-order to guarantee maximum availability of data to be stored in MU.

The Working Model of Mobile Clients: In Fig.9 the Mobile Unit (MU), requests the data from the local resources through the client cache manager. If the cache manager respond, then the process will be done locally, otherwise the cache manager will contact the source Mobile Server (MS).

Pseudo code:

If accessed data = "in-local-cache" then

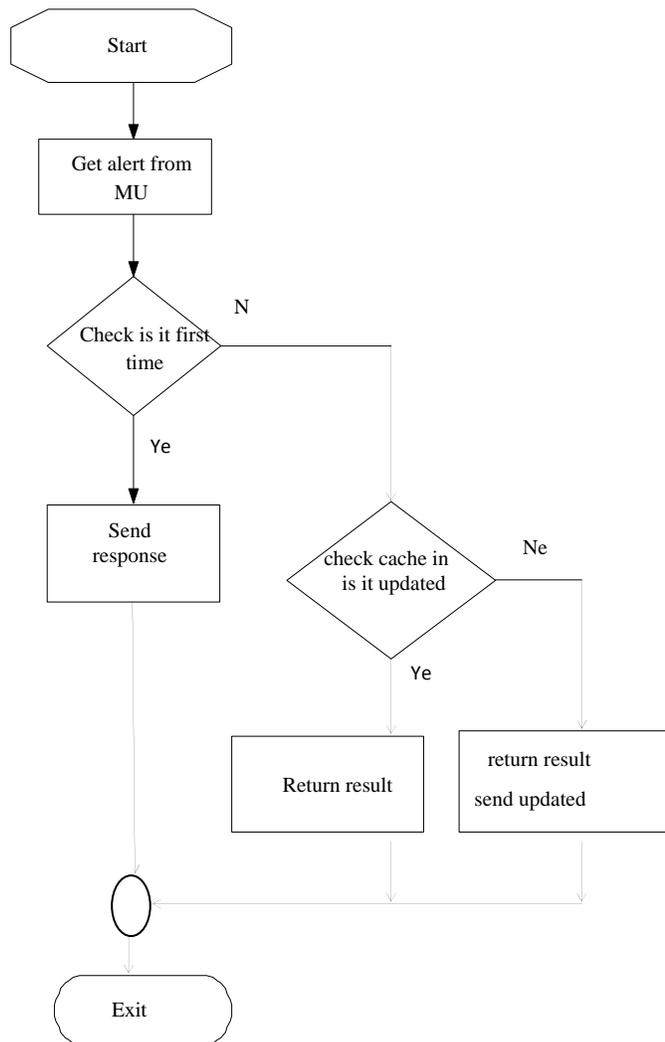
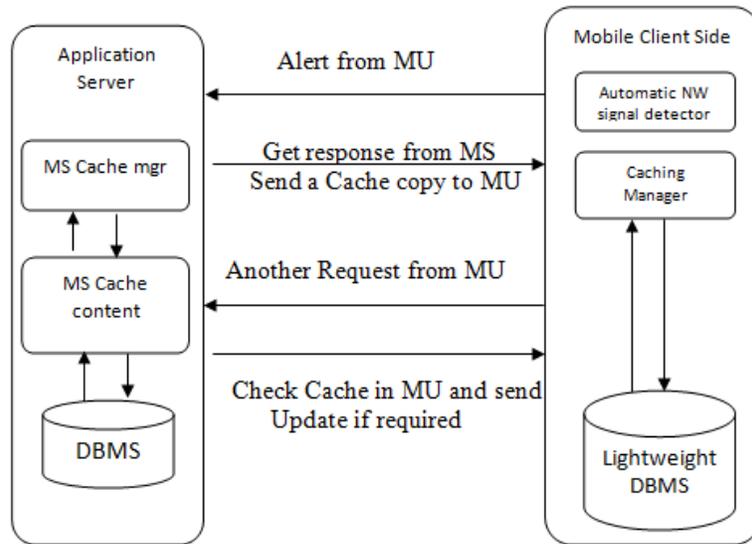


Fig. 10 Base Server Model

```

Send-response
Else
If accessed data = "not-in-local-cache" Then
Call cache-mgr
Send-query-to-server
If query-response-from-server successful Then
Save-copy-to-cache
Else
Send error-notification to the client
End If
    
```

In fig. 10, the Mobile Server (MS) processes the requests from the Mobile Client (MU), as explained previously. When the MU issues the query, the MS will respond from the local original data and then it will check if it's the first visiting for the MU. If so, the MS will send the query answer to the MU. If not, then the MS will check the cached data in the MU whether it is up to date or not. If it's up to date, then nothing will happen, otherwise, an update will be sent to the MU along with the notification.

Pseudo code:

```

If data-query-alert = "first-time" Then
Send-response-to-mu
Else
If data-query = "second-time" then
Check-update
If old-data = "need-update" then
Send-update-to-mu
Else
Abort notification to mu
End If.
    
```

CONCLUSION

Mobile technology is growing rapidly day by day. Now we are a witness of a huge plethora in using mobile devices for conducting business transactions that involve data from a remote database. But there are many issues that arise when using mobile clients in business such as network connectivity, power consumption and low bandwidth. In this study we consider the mobile data caching to solve the problem of data unavailability caused during network disconnection. The contribution of this study is to develop a comprehensive strategy for caching and data management in mobile clients. This strategy will enable mobile clients to work with the maximum data availability even during off line or disconnected mode. The maximum availability will enhance the query response time as well as reduces the network traffic which in turn will result in saving the network bandwidth.

REFERENCES

ARI, I.A.G.L.M.A.B.D.E.L., 2002. ACME: Adaptive Caching Using Multiple Experts. *Proceedings in Informatics*, 14.

Ahmed, A., Dominic, D.D. & Abdullah, A., 2011. A NOVEL REPLICATION STRATEGY FOR LARGE-SCALE MOBILE DISTRIBUTED DATABASE SYSTEMS. *Science And Technology*, 6(3), pp.268 – 299.

Anandharaj, G. & Anitha, R., 2009. A Distributed Cache Management Architecture for Mobile Computing Environments. *2009 IEEE International Advance Computing Conference*, (March), pp.642–648. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4809087>.

Artail, H., Safa, H. & Pierre, S., 2005. Database caching in MANETs based on separation of queries and responses. *WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.*, 3, pp.237–244. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1512909>.

August, B., 2009. An architecture for data synchronization in a mobile environment., (August).

Barbara, D. & Imielifiski, T., 1995. Sleepers and Workaholics: Caching Strategies in Mobile Environments (Extended Version). *Vldb Journal*, 602, pp.567–602.

Cao, G., 2003. A scalable low-latency cache invalidation strategy for mobile environments. *IEEE Transactions on Knowledge and Data Engineering*, 13(1251-1265).

Chauhan, N., Awasthi, L.K. & Chand, N., 2011. Data Caching with Intelligent Prefetching in Mobile Ad Hoc Networks. *2011 International Conference on Communication Systems and Network Technologies*, pp.71–75. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5966407> [Accessed December 6, 2011].

Das, G.C.L.Y.C.R., 2004. Cooperative Cache-Based Data Access in Ad Hoc Networks. ", *IEEE Computer Society*.

Ding, Z., Mengt, X. & Wangt, S., 2001. A Novel Conflict Detection and Resolution Strategy Based on TLRSP in Replicated Mobile Database Systems*. *Queue*, pp.234–240.

Du, Y., & Gupta, K.S., 2005. COOP: A cooperative caching service in MANETs. . *In Proceedings of ICAS-ICNS*, pp.58–63.

Fageeri, S.O., Ahmad, R. & Mu, A.A., 2012. Mobile Data availability Similarity between Cooperative Caching and Replication. *IOSR Journal of Computer Engineering (IOSRJCE)*, 1(4), pp.9–13.

Fritzke Jr, U., R. P. Valentim, 2008, Adaptive Replication Control based on Consensus Categories and Subject Descriptors. *DBMS*, pp.1-10.

He, C.C. and X., 2006. Research on application of caching technology in mobile databases. *Computer Engineering and Design*, 3.

- Itani, Z.2005, Efficient Pull Based Replication and synchronization for Mobile Databases. *Access*, pp.5–8.
- J.Abawajy, M.D. and M.O., 2006. novel data replication and management protocol for mobile computing systems. *Mobile Information Systems*, Volume 2(Number 1/2006), pp.3–19.
- Cho, J., S. Oh, et al., 2003. Neighbor caching in multi-hop wireless ad hoc networks. *IEEE Communications Letters*, 7, pp.525 – 527.
- Khairullah, E.F., 2009. Improving Accuracy of Context Aware Data Replication In Mobile Computing Application. *Context*, pp.498–503.
- Kumar, S. & Mohania, M., 2002. Mobile data and transaction management. *Information Sciences*, 141, pp.279–309.
- Meng Xiaofeng Wang Shan Zhiming, D., 2002. A transactional asynchronous replication scheme for mobile database systems. *Journal of Computer Science and Technology*, Volume 17.
- Mershad, K. & Artail, Hassan, 2009. Semantic Caching for Mobile Ad Hoc Networks. *2009 Fifth International Conference on Mobile Ad-hoc and Sensor Networks*, pp.25–32. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5401539> [Accessed November 28, 2011].
- Monteiro, J.M., 2007. A Mechanism for Replicated Data Consistency in Mobile Computing Environments. *Computer*, pp.914–919.
- Myers, M.C.A.A.B.L.A.C., 1997. HAC: Hybrid Adaptive Caching for Distributed Storage Systems. *Proceedings of the 14th ACM Symposium on Operating Systems Principles*.
- Narottam Chand, R.C.J. and M.M.2006, Efficient Cooperative Caching in Ad Hoc Networks. *IEEE Transactions*, pp.1–8.
- Narottam Chand, R.C.J. and M.M., 2006. Efficient Cooperative Caching in Ad Hoc Networks. *Transactions., IEEE*, pp..1–8.
- Rathore, R., 2008. An Overview of Mobile Database Caching. *unpublished*, pp.1–15.
- S. Lim, W. Lee, G. Cao, C.D., 2006. A Novel Caching Scheme for Internet Based Mobile Ad Hoc Networks Performance. *Ad Hoc Networks*, vol. 4, no, pp.225–239.
- Sovran, Y.,2011. Transactional storage for geo-replicated systems. *October*, pp.385–400.
- Starner, T., 2003. Powerful change part 1: batteries and possible alternatives for the mobile market. *IEEE Pervasive Computing*, pp.2(4):86–88.
- Tarafdar, M. & Haghjoo, M.S., 2010. Location Privacy in Processing Location Dependent Queries in Mobile Database Systems. *Computer Engineering*, pp.181–186.
- Tian, J. & Denko, M.K., 2007. Exploiting Clustering and Cross-Layer Design Approaches for Data Caching in MANETs. *Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2007)*, (WiMob), pp.52–52. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4390846>.
- Tolia, N., Satyanarayanan, M. & Wolbach, A., 2007. Improving Mobile Database Access Over Wide-Area Networks Without Degrading Consistency. *ACM978*, pp.71–84.
- Vakali, A., 2001. Proxy Cache Replacement Algorithms: A History-Based Approach. *World Wide Web*, 4(4), pp.277 – 298.
- Vora, A.A., Technology, I. & Portfolio, T., 2005. Data Stashing Strategies for Disconnected and Partially Connected Mobile Environments. *Networks*, (June).
- Wang, Y. *et al.*, 2008. Caching Invalidation Strategies for Supporting ' Weak ' Location Dependent Queries. , pp.459–464.
- Yin, L. , A.H. *et al.*, 2006. Supporting cooperative caching in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(1), pp.77–89. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1542018>.
- Zaslavsky, A. *et al.*, 1996. Primary Copy Method and its Modifications for Database Replication in Distributed Mobile Computing Environment. *Read*, pp.178–187.