

A New Replication Strategy for Unforeseeable Disconnection under Agent-Based Mobile Computing System*

Keith K. S. Lee and Y. H. Chin
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 300, R.O.C.
Email : {kslee,yhchin}@cs.nthu.edu.tw

Abstract

The goals of this paper are to revise previous replication strategies and propose a new hierarchy structure, client-agent-server architecture, to fit a mobile computing system that has more resource constraints than a conventional distributed database system has. This three-tire agent-based mobile computing system frees the limitations on time and whereabouts of computing. Moreover, a new replication strategy, three-copy replication that is based on optimistic replication strategy, is proposed to deal with the problems of mobility management and disconnection management in a mobile computing system. It allows mobile clients to read and update the database while they are disconnected from the network, if the mobile copy is available. In addition, the implementation issues of this new replication strategy are also provided.

Keywords: Data replication, mobile computing, distributed database, constraint checking, fault tolerance.

1. Introduction

Recent advances in hardware and communication technology have made mobile computing possible. It is expected that, in the near future, tens of millions of users will carry a portable computer with a wireless connection to a worldwide information network. This rapidly expanding technology poses some new challenging problem [9].

The position of wireless networks relative to wired networks can be thought as an extension to wired networks. As we can see, the applications of mobile computing are increasing every day. With wireless, the mobile user, such as a salesperson, can spend his time in the office, in his home, or at a public area such as a shopping mall, a downtown area, or somewhere in between such as in his car, on a plane, etc. Before arriving at the customer's location, the salesperson can perform some functions such as sales quotation, inventory check, order entry, and/or

credit authorization in above areas, while he is connected to servers back at his home office, or to any database he needs to perform his job. It was predicted that for mobile computing applications by the years 2005 [6], almost half of the people using mobile computing will be performing mobile office applications such as electronic mail, facsimile, file transfer, and database access. In addition, about a quarter, the mobile user will use it for personal communication such as message delivery, calendar arrangement, and directory services. Finally, the rest of mobile user using mobile computing will be performing such as transportation, field sales, and field service.

We argue that a number of implicit assumptions on traditional distributed system are not reasonable in a mobile computing system. In a traditional distributed system, for example, the location of various hosts is well known and the communication cost of various hosts is symmetric. The connection link usually is a high-bandwidth connection, and the scale of system is normally low latency. Moreover, there are no power restrictions and data storage space on the hosts is general huge. However, in a mobile computing system, due to mobility, the location of the hosts is dynamically changed. The communication between stationary hosts and mobile clients is asymmetric. That is, the communication cost of up-link stream of a mobile client is more expensive than that of down-link stream of a stationary host via the wireless media. Mobile clients generally have limited bandwidth for communications and high latency. The battery power of a mobile device is a limited and precious resource. The stable data storage space is no longer necessarily available on a mobile client. These new challenging issues must be investigated while considering the replica control strategies in mobile computing with mobile clients.

Due to the mobility of users and services, replication schemes will be affected by mobile computing environments. Moreover, data replication has long been used in a distributed database system to increase data availability and reliability, to improve system performance, and to maximize network bandwidth utilization. How can we apply data replication techniques using in a distributed database system [4] into mobile computing environment to

* This work was partially supported by the Republic of China National Science Council under Contract No. NSC 87-2213-E-007-002.

overcome the nature resource limitation of mobile clients? First of all, we propose a three-tier architecture, client-agent-server architecture, which will be explained later. Agents play an important role of data management so that a mobile client can gain the following profits: power saving, bandwidth saving, continuous processing, and consistency maintenance controlling. Moreover, in such a hierarchy environment, we provide a new replica control strategy, three-copy replication scheme, so that a mobile client can continuously process the required task even when he is disconnected from the network.

The replica control strategies in a distributed database system cannot directly apply to the mobile computing system, due to the mobile clients are often disconnected from the network for the sake of power saving or out of radio coverage, even intermittent or weak connection. So, disconnection is a part of characteristics in mobile computing rather than a failure mode. If the mobile clients can cache some useful data properly, they can continuously operate on local cache data; consequently more work can be done during disconnection. There are three types of operations that are suited for disconnecting mobile clients. First operation is on a disjoint of the actual data. For example, each mobile client is allocated a subset of available tickets. Therefore, they can independently sale to customers under the number of local available ticket constraint rather than the global available ticket constraint. Hence, the communication cost of wireless link between a database server and a mobile client is saved in this case. It satisfies the disconnection operations requirement of a mobile client. Second operation is on increment or decrement data. The value of replication data is not beyond the limit value of available data. Third operation is on close-held data. That is, the replication data is used frequently in mobile client and is local satisfaction. Then, for a disconnecting mobile client, the number of data conflicts on these operations shall be as less as possible. In this paper, some conflict checking strategies are proposed to deal with potential data conflicts. Once detected, the resolution will involve both automated and manual procedures.

There are two families of replica control strategies, pessimistic and optimistic, that can be used to deal with the disconnection operation. A pessimistic replication is a lock-based replication. It synchronizes all replicas at all nodes and then updates all the replicas as part of one atomic transaction. On the other hand, an optimistic replication is an asynchronous replication. The updated transaction is committed first, then asynchronously propagate replica update to other corresponding nodes. In this paper, we propose a more flexible model based on optimistic replication strategy. We argue that the database interface need be modified via adding some weak operations such as strict read and loose read operations. Weak operations support disconnected operation since a mobile client can operate disconnectedly as long his applications are satisfied with local copies. Data are stored or cached at a mobile client to support its autonomous operations during disconnection and to avoid the connection cost of the network. [KS92] also address optimistic replication strategy providing the highest possible availability of data that better match our design goal. In this paper, our previous work on replication strategy is extended to be primary copy, secondary copy, and mobile copy, we called *three-copy replication* scheme,

to deal with the possible disconnection.

Definition 1.1 [mobile copy]. Either by system detection or user specified, there is potential disconnection occurrence of a mobile client. A foreign agent will replicate the primary copy from a database server as the secondary copy, then replicate it to the mobile client as the mobile copy with log records and time stamp of corresponding replicated data.

Once the mobile client is disconnected from the network, local cache (e.g., mobile copy) preserves continuous operations. Any update from the database server will reflect to secondary copies at the foreign agent first. Then the foreign agent will broadcast the invalid information under the cell. In a doze mode of the mobile client, the wireless device on the mobile client will store the information. While the mobile client awake to ready to receive message, it will find the mobile copy is modified. Consistency between the secondary copy and the mobile copy must be maintained. On the other hand, in a disconnection mode the mobile client is not available, the foreign agent stores the update information until the mobile client is reconnected or another foreign agent is performing the service handoff procedure to inform the current foreign agent.

We assume each mobile client up-links periodically or non-periodically its user profiles to its foreign agent. In addition, a foreign agent maintains a data structure to record the status of mobile clients that are served by the foreign agent. In general, the foreign agent will perform the following functions:

- To handle local conflict; if there are more than one mobile client under the same agent, it will perform conflict checking.
- To perform information filtering; the network is distinguished as a wire from a wireless part. Only the wireless traffic will be transferred via the agent. The wireless traffic is corresponding to the mobile clients under the foreign agent, Moreover, the foreign agent will perform information collection to reduce the wireless traffic load.
- To maintain data consistency while a mobile client disconnection;
- To forward message for a mobile client; an agent creates a deferred list of replica updates and sends this list while a mobile client is reconnection.
- To execute power management for a mobile client.

The organization of the remainder of this paper is as follows. Section 2 introduces the related work of replication strategies in mobile computing system. In Section 3, we propose a model of agent-based mobile computing system. A new replication strategy is addressed in Section 4. In Section 5, we show the implementation issues and further discussion. Finally, Section 6 presents conclusions and future work.

2. Related Work

In a wire network, disconnection is considered as a link failure. However, in a wireless network, disconnection may not due to a link failure. For example, for the sack of

saving the connection cost and the network bandwidth, we often write electronic mail on the disconnection mode, then send it to the destination on the connection mode.

Disconnection can be classified into two categories: predictable disconnection and unforeseeable disconnection. Using clustering method according to the semantic of data (e.g., location data) and user profiles provided by mobile clients, predictable disconnection can be solved. It was proposed in [5]. They dynamically move the disconnection mobile client from a consistency class to a divergence class. Once the disconnection mobile client is reconnected from the networks. It will be moved from a divergence class to a consistency class. To our best knowledge, however, unforeseeable disconnection problems do not appear in any literature. In this paper, we will focus on the problem in a mobile environment and propose some feasible solutions.

A clear overview of data replication is provided in [2]. They specify the dangers to apply replication technique to a mobile environment. When the workload scales up due to data replication and mobile client mobility, the probability of deadlock or reconciliation will increase dramatically in the replication model of update anywhere-anytime-anyway. However, they do not consider that proper data arrangement will reduce conflict violations and the issues of power management of mobile clients. In [7], they address the cache invalidation methods that are suited for an agent-based mobile computing system. We apply this method into our asynchronous replication model. In the case of push-based model, which is the one-to-many replication of reference data out to the mobile clients, is used for the simple one-way replication of read-only reference data to all mobile clients. While the reference data is updated on a database server, according to the requirement of mobile clients, a cache invalidated report will be send from the database server to the corresponding mobile clients to inform the update. Then mobile clients can renew their mobile copy on time (strict read) or lately (loose read). On the other hand, if the mobile client is disconnected from the network in the period of a cache invalidated report being sent. Then its agent will cache its report until it is reconnected into the networks and is informed by its agent. However, in our approach, the invalidated report is sent in asynchronous manner rather than in synchronous manner, according to the user profiles that are specified by mobile clients. Our asynchronous approach is different from [7] that is using synchronous approach.

In [1], according to read and write requests, they consider the one-copy allocation scheme and the two-copy allocation scheme. If a mobile client issues the number of read operation requests more than that of write operation requests. Then, the mobile client can hold the replication data. That is the two-copy allocation scheme. On the other hand, if a mobile client issues the number of read operation requests that are less than that of write operation requests. Then, the mobile client will not hold the replication data. That is the one-copy allocation scheme. However, they did not consider the case of disconnection. In the case of unforeseeable disconnection, the two-copy allocation scheme will be the best allocation scheme, because there always has a local copy for the purpose of local process. That is why we propose three-copy replication to support the disconnection case. Both wire and wireless part of network at less have one copy in each

part. That is, there exist one copy at a database server and one copy at a mobile client. Moreover, there exist one copy at an agent for mobility management and disconnection management handling.

3. System architecture

In our previous work [4], we proposed a loosely coupled distributed database system in which each database server is able to operate independently. In such an environment, a set of database servers interconnected via a wire network to provide aggregate database services. Each server is capable of storing and manipulating a limited amount of data. However, a user can issue a query or an update request based on the entire data set to any server. If the required data resides entirely on the database server, this request may be completely served locally. Otherwise it has to be shipped to other servers for processing. That is, a model of distributed database computing consists of a set of fixed clients and fixed database servers. However, in a mobile computing environment, due to the mobility of client and/or server with wireless communication capability, we extend our previous system as client-agent-server architecture to be a model of mobile computing to overcome the problems of mobility and disconnection management. We call such a system *agent-based mobile computing system* since each agent is able to handle the wire and wireless traffics for clients and database servers properly. Its architecture is depicted in Figure 1.

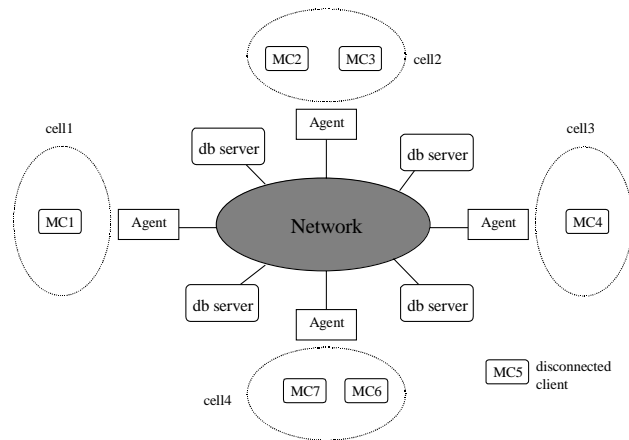


Fig. 1. Agent-Based Mobile Computing System

There are three possible architectures of agent-based mobile computing system as follows: one is mobile client, fixed foreign agent, and fixed database server [8]. For example, a salesperson (mobile client) via his local sub-company (fixed foreign agent) to access the information on his home-company database server (fixed database server) anytime and anywhere. Our work is focus on this framework. One is mobile client, fixed foreign agent, and

mobile database server. For example, a salesperson (mobile client) on his vehicle via microwave station (fixed foreign agent) to access the satellite (mobile database server) information for some satellite applications. Last one is mobile client, mobile foreign agent, and mobile database server. Such a system is an ad hoc network that is a collection of wireless mobile clients forming temporary network without the aid of any established infrastructure or centralized administration. For example, each salesperson with mobile device can directly share the information to each other.

In general, in an agent-based mobile computing system, a mobile client requests services such as database access. In a connected mode, a mobile client can issue local or/and remote operations. On the other hand, it can issue local operation only in a disconnected mode. Database servers provide data services and handle global conflict. In addition, an agent provides the following functions: handle local conflict and perform information filtering; maintain data consistency while a mobile client disconnection; forward message for a mobile client; and execute power management for a mobile client. Why we need agent approach? From the standpoint of data cache, an agent can cache some information that was visited by others such as the proxy in WWW. A client can retrieve the required home page from the near-by agent rather than from the home page servers that may far away from the mobile client. That is, the communication time between the database server and the mobile client will be reduced. Another reason is that agents can cache some useful information, such as invalidated report, for mobile clients, while they are disconnected from the network.

3.1. Conflict checking

We assume the data cached on a mobile client are subset of that cached on an agent to keep the size of the mobile reference data small. While the cached data is updated on a database server, a mobile client will be noticed via an agent. Global conflicts should be handled via the database server. Consider the following situation in figure 1; the mobile client 1 (MC1) and the mobile client 2 (MC2) are under the different agents and the same database server. Only the common database server knows what data are replicated on which mobile clients. However, Local conflicts can be handled via the foreign agent. Consider the following situation: both the mobile client 2 (MC2) and the mobile client 3 (MC3), in figure 1, are under the same agent and database server. So, the consistency problem can be handled via the common foreign agent.

More detail, there are two types of update conflicts: one is intra-table update conflict which are detectable within the scope of a single table; the other is inter-table update conflicts which are not detectable within the scope of a single table, but which are nonetheless conflicting. We apply intra-table update conflicts checking as local conflicts checking into a foreign agent and apply inter-table update conflicts checking as global conflicts checking into a database server.

3.2. Mobility management

An agent-based mobile computing system supports the

mobility of mobile clients. While a mobile client move from one cell to another cell, the following procedure will be proceeded:

1. Agent discovery

Firstly, the mobile client executes the basic handoff that automatically registers with the nearest base station. Then the mobile client submits an agent discovery procedure to the corresponding base station. We assume that there exist one or more base stations in one cell. The base stations have the capability of receiving signal from and transmitting beacon to mobile clients. The procedure of agent discovery is as follows. The mobile client can send out a solicitation packet asking if a foreign agent is in the vicinity. When the corresponding foreign agent receive that packet, it will reply that "I am a foreign agent". The mobile client can decide which one is the candidate foreign agent. Another possible and simple way is through advertisements. Foreign agents send out service advertisements that announce their willingness to provide services to visiting mobile clients. Such an approach will increase the load of the networks and its own resources.

2. Service handoff

After a new foreign agent is found, the previous foreign agent forwards the cache data and replicates mobile client's personal profile status to the new foreign agent. Transaction execution status is also included. For turn around mobile client, the state of this mobile client will be kept on both agents. Then, the local and global conflict checking procedure will be invoked via the new foreign agent. Through service handoff, the new foreign agent always knows the locations of the database server of mobile clients. So, Global conflict checking can be invoked in the proper database servers.

3. Store and forward paradigm

A store and forward paradigm makes the disconnection of a mobile client management more efficient. In store phase, data will be stored as a deferred list. In a doze mode, the wireless device of a mobile client stores the messages. In a disconnection mode, the mobile client is not available, the foreign agent stores the messages. In forward phase, the stored message can be forwarded to the mobile client while it reconnects to the networks.

3.3. Power management

We discuss how to provide power management features in an agent-based mobile computing system to help mobile clients that are running on batteries and must sleep as often as possible. The way of power saving scheme is under the help of a foreign agent. As explained in the introduction section, a foreign agent knows the power management states of the mobile clients that are associated with it. In addition, a mobile client is always connected with the nearest foreign agent so that the waiting time of a request response is minimal. This also can save the battery power of mobile clients. The power management states of mobile clients on foreign agent are

as follows:

- Transmit: The transmitter of a mobile client is on.
- Awake: The receiver of a mobile client is on.
- Doze: Both the transmitter and the receiver of a mobile client are off. The timer may be on.
- Disconnection: A mobile client is out of radio coverage, wireless link failure, or for power saving reason power off.

The state of power management of each mobile client is recorded on the foreign agent. According to these records, the foreign agent can make decisions to transmit information or not in the proper time. The consumption of battery power of a mobile client from large to small is the transmitted state, the awaked state, the dozed state, and the disconnected state. Through the foreign agent, a mobile client can stay in the low-power mode as long as possible so that the battery power of a mobile client can be saved. First of all, the foreign agent will buffer traffic for dozing mobile clients. So, the mobile client may be in doze mode as long as possible. Secondly, The invalidation information, included in each beacon indicates traffic destined for dozing mobile clients, will be sent from foreign agent to mobile clients in the same cell. Dozing mobile clients wake up periodically to listen to the invalidation information in the beacon. The period between wake up to sleep and sleep to wake up is adjusted in a mobile client, according to the mobile client specified strict read or loose read and the degree of loose read. The foreign agent then sends an index information followed by data for that mobile client. The foreign agent transmits a beacon that includes the following information: time stamp, beacon interval index period, index count, etc. That is, the beacon includes which mobile clients have data to receive, how much data to receive, and when it will be delivered.

4. Three-copy replication scheme

This section presents the method of using data replication to serve a mobile client continuous operation while it is disconnection from the network. We assume that a database server and a foreign agent are both on the fix part of networks as depicted in figure 1. The fixed primary copy is on a database server and the fixed secondary copy is on a foreign agent. Both the database server and the foreign agent have unlimited and stable data storage. They handle a large amount of update transactions. They are no power restriction and have lower communication cost. However, in more general case, a mobile client can be the primary data owner, will be addressed in the next further discussion section.

In an agent-base mobile computing system, a home database server owns the primary copy of data, a foreign agent owns the secondary copy of data, and a mobile client owns the mobile copy of data. We call the primary copy, the secondary copy, and the mobile copy as *three-copy replication* scheme. For example, business travelers use mobile computers to enable them to work and to access data on the road. Data at their company can be the primary copy. While they go out for business, local agent will be found. Data buffering at agent from their home company can be the secondary copy. Finally, they can cache the subset of data of agent on their notebook for local usage to

keep the size of the mobile reference data small. Data at the mobile client can be the mobile copy.

In our three-copy replication approach, transactions can be executed on a database server and a mobile client independently. However, the data conflict violations will be keep minimal, if data is arranged properly. This will be discussed in the next section. Assume that database server own the primary copy of data. On connection mode, all the update operations on the primary copy have to be sent to the database server. Or, some update operations are issued via an unforeseeable disconnection mobile client. These operations will be cached on the mobile client temporarily. Then, the update effects on database server are propagated to other replicas on the foreign agents and mobile clients asynchronously. If some mobile clients are disconnected from the networks, their foreign agent will cache these update effects as deferred list and apply to the secondary copy first. Until these mobile clients reconnect into the networks, the secondary copy on the foreign agent will be propagated to mobile clients and renew the mobile copy on the mobile clients. This ensures multiple writes do not take place at the same time and the active transactions do not have to wait until all other replicas are updated. On the other hand, on disconnection mode, the update transactions will be executed on the mobile copy first to support continuous operations. Like log-base scheme, the update transactions on a mobile client will be executed later on the corresponding database server. Once the mobile clients reconnect to the networks, they must re-synchronize the mobile copy and the primary copy through the secondary copy.

For re-synchronization steps while mobile clients change connection mode from disconnection to be connection, there are four cases must be considered. Case 1, before reconnection, there are update operations at the primary copy and no update operations on the mobile copy. Once the mobile clients reconnect to the networks, it can renew its copy with the primary copy by updating the changes made to the primary copy via the secondary copy on an agent. To save the power of a mobile client, the corresponding secondary copy on the agent will be refreshed firstly. Then the renew process of a mobile client will be handle by the nearest agent using this refreshed secondary copy instead of the primary copy on database server which may far away from the mobile client. Therefore, the battery power of a mobile client can be reduced, because the minimum of time in transmit state is persisted.

Case 2, there are update operations both at the primary copy and the mobile copy before reconnection. But, according to the time stamp of transaction log, the update behavior on the primary copy is before on the mobile copy. We assume that compensation transactions could rollback the committed transactions of the mobile clients. Compensating transaction is performed to ensure data consistency. The mobile client will roll backs its transaction and updates the mobile copy with the latest version of primary copy from the database server and then it re-executes its previous transactions. In an asynchronous replication environment, use of compensating transactions means that sometimes the updates of a transaction are not durable. Loss of durability creates a rippling effect in that the transactions that are subsequently undone were viewed by other transaction before the undo process occurs.

Case 3, the update behavior on the mobile copy is

before on the primary copy. To reduce the inconsistency, the mobile client must inform the database server as soon as possible. Then, the database server will reflect these effects on the primary copy. As we discussed in previous section, the up-link traffic is more power consumption than the down-link traffic, because the mobile client is on transmit state instead of on awake state. Moreover, this informing procedure will pay the price of performance and cost benefits of disconnection. However, this policy will minimize the out-dated-read problem. On the other hand, two different read semantics, e.g., loose read and strict read, are provided in order to reduce the down-link traffic. There is a tradeoff between communication cost and consistency status. Case 4, the update transactions occur separately on the mobile copy and on the primary copy at the same time. This conflict situation must be solved by manual interaction.

5. Implementation issues and discussion

In this section, we will discuss the implementation issues of asynchronous replication and constraint checking to make *three-copy replication* scheme feasible. We compare the two-tier architecture with three-tier architecture by their advantages and disadvantages.

5.1. Comparison two-tier architecture with three-tier architecture

The main advantages of using the two-tier client-server architecture include the following:

- Simpler to implement
- Faster to development
- Less expensive to run

The principle disadvantage is the functional limitation and inflexibility of the DBMS stored procedure mechanism. An example is that a stored procedure is limited to one DBMS. Another disadvantage is that it does not suit for the case wireless network existence. The database server will be a traffic bottleneck while the population of mobile clients are variety quickly [2]. However, the workload can be shared via the agents. In other words, an agent can partition the whole networks as fixed networks and wireless networks. It can take care of the traffic and function on wireless networks.

By contrast, the three-tier client-agent-server architecture is more flexible and vigorous because the application logic does not run under the control of a DBMS. Control is usually provided by some flavor of middle-ware on an agent and a database server. The main advantages of using the three-tier architecture include the following:

- Sharing the workload of database server by agents such as constraint checking and data caching.
- Handling the location management of mobile clients by agents. According to the user profile of a mobile client, an agent can find the location of a mobile client more efficiently.
- Saving the power usage of the up-link traffics of a mobile client. An agent plays an intermediate role between a mobile client and a database server. The

communication time via wireless link can be reduced significantly.

- Supporting unforeseeable disconnection of a mobile client. The mobile client can continuously process using mobile copy of data.

5.2. Minimum conflict violations by proper data arrangement

As we discuss in the introduction section, there are three types of operations that are suited for disconnecting mobile client. Through proper data arrangement, the conflict violations will be keep minimum.

- Operation is on a disjoint of the actual data.
- Operation is on increment or decrement data.
- Operation is on closely-held data

Two simple asynchronous replication models are used to implement the data arrangement. The first is a *push-based model* that is the one-to-many replication of reference data out to the mobile clients. This is used for the simple one-way replication of read-only reference data to all mobile clients. On a database server, update and read operations can be issued on the primary copy and its local data. In addition, on the mobile client, update operations can be issued only on the local data and read operations can be issued on both the mobile copy and the local data. Once the primary copy on a database server is updated, the invalidated report that contains the time stamp of the latest change data will be sent. According to the user profile of a mobile client provided, a mobile client specifies the data tolerance in it. A database server will transmit the invalidated report and broadcast via corresponding agents in the proper period of time. Once mobile clients receive the invalidated report, they either purge it from the cache or update the cache's time stamp to the time stamp of the report. In our approach, the invalidated report is sent in asynchronous manner.

The second is a *pull-based model* that is the many-to-one replication of transactional data back to a consolidated whole. This is used to collect all data entered from mobile clients into a database server. That is, on each mobile client, update and read operations can be issued on its local data as the mobile copy. In addition, on a database server, only read operations can be issued on the primary copy and read operations can be issued on both the mobile copy and the local data. Fault tolerance can be provided in the pull-based mode. Assume that a mobile client occur failures such as a corrupted disk device. Because the mobile client has replica on a database server as the primary copy, its data can be reconstructed from the primary copy on database server. In essence, the reconstruction procedure provides a data source that is closely consistency with the original mobile copy. It can be used to replace the mobile copy in the event of failures at the mobile client. Each model is orthogonal to the other in that each supports different data elements. If need be, both of these asynchronous replication processes can occur during the same communication session.

The following address a more general and complex design pattern. It is considered more generally, because a mobile client may own some local data as primary source that can be replicated into other computers in the networks

and plays a role of a database server. It is considered more complex because it involves two-way replication or requires the migration of the primary source across multiple replicas. This model is generally used where update activity is restricted to the data that is locally owned but where querying and/or reporting requires a total view. The complexity of this model is not in its implementation, but in its maintenance. So, some conflict detection and resolution strategies are needed. For this case, by partitioning and assigning ownership, the number of conflicts can be kept to a minimum. Data element ownership is important because it will be used to determine which updater wins when conflicts are detected. We assign ownership table that is incorporated into the database. This table identifies the primary source owner. Moreover, a latest updater column and a latest updated time stamp column are incorporated into every table that will reside within the mobile client database. Before updating, the ownership table will be lookup. If a mobile client is updating outside of the scope of ownership, this transaction will be completed first. As part of that transaction, the entry is flagged as a potential conflict. Once the potential conflicts are detected, business rules are applied to attempt resolution. If some potential conflicts cannot be immediately resolved, then they are written as an exceptions file that is resolved manually.

5.3. Constraint checking

Asynchronous data replication provides what is called loose consistency between different copies. This means that the latency before data consistency is always greater than zero; the replication process occurs asynchronously to the originating transaction. In other words, there is always some degree of lag between the time for committing the primary copy and the time for availability of replica.

There are two problems must be overcome while using asynchronous data replication. One is partial results problem. The other is referential integrity constraints violation problem. We use the following example to describe the above problems. At the primary copy, a bank originating transactions are Transaction1 and Transaction2. Transaction1 = {Begin; update table A; update table B; update table C; update table D; Commit} and Transaction2 = {Begin; update table A; update table B; update table E; update table F; Commit}. The target data of a mobile client are the balanced account. For asynchronous data replica, transactional semantics is not preserved. With this type of replication the onus for the integrity of the data at target replicas falls totally on the scheduling of the updates for each target. We assume that Transaction1 and Transaction2 share the same originating data source. Target replica(s) can be applied the following transactions: {Apply updates to table A commit}, {Apply updates to table B commit}, {Apply updates to table C commit}, {Apply updates to table D commit}, {Apply updates to table E commit}, and {Apply updates to table F commit}. If tables A, B, E, and F are updated as one replication set at the time T1. And tables C and D are updated later at the time T2. During the time between T1 and T2, mobile clients see all of the results of Transaction2 and one-half of the result of Transaction1. This is partial results problem. This may or may not be acceptable from a business perspective.

A further complication of not preserving transactional semantics during the replication process is the potential for failed updates at a target replica due to referential integrity constraints defined at the target. These violations occur because the appliance process commits updates to only one replica table at a time; it does not preserve a transactional whole across replica tables. That is inter-table update conflicts as described in Section 3. In this situation, the table order within the appliance process becomes critical. This is referential integrity constraints violation problem.

6. Conclusions and future work

We have proposed an agent-based mobile computing system that mobile clients can access data of the database servers anytime and anywhere, even if they are disconnected from the network. In such a mobile computing system, an agent plays an important role for helping mobile clients the following work: handle local conflict and perform information filtering; maintain data consistency while a mobile client disconnection; forward message for a mobile client; and execute power management for a mobile client. We have compared the two-tier client-server architecture with three-tier client-agent-server architecture. The advantages of our agent-based mobile computing system are more flexibility and usefulness for handling mobility and disconnection of mobile clients.

A new replication strategy, three-copy replication scheme, is proposed for unforeseeable disconnection mobile clients that can local process their transactions under the disconnection situation. Four cases studies is addressed for consistency maintenance using time stamps, deferred list, and compensating transactions. Moreover, We have discussed the implementation issues of three-copy replication scheme. Push-based model and pull-based model is provided for asynchronous replication. In addition, we have clarified there are two important problems, partial results and referential integrity constraints violation problems, while using asynchronous data replication.

In the future, we will evaluate the performance issues of three-copy replication scheme. Consider the following trade-off perspectives. One is local process on cache data of a mobile client. One is remote process on a database server. Then the results are transmitted from the database server to the mobile client via an agent by wireless link. And one is concurrent process the same transaction on both fixed networks part and the mobile client. We will consider under what condition and case will be the best. The performance issues are studied from the standpoints of following viewpoint: processing power of mobile client, communication capability of mobile client, storage issue, and consistency issue. Moreover, we will combine three-copy replication scheme with a location management strategy, which is called *adjustable location method* that is current work in another paper, based on history prediction to avoid entire search the wireless system and reduce the total search cost.

References

- [1] Yixiu Huang, Prasad Sistla, and Ouri Wolfson, "Data

- Replication for Mobile Computers,” In *Proceeding of the ACM SIGMOD 1994, International Conference on Management of Data*, pages 13--24, 1994.
- [2] Jim Gray, Pat Helland, Patrick O’Neil, and Dennis Shasha, “The Dangers of Replication and a Solution,” In *Proceeding of the ACM SIGMOD 1996, International Conference on Management of Data*, pages 173--182, 1996.
- [3] J. J. Kistler and M. Satyanarayanan, “Disconnected Operation in the Coda File System,” *ACM Transactions on Computer Systems*, 10(1):3--25, February 1992.
- [4] S. Y. Hwang, Keith K. S. Lee, and Y. H. Chin, “Data Replication in a Distributed Database Systems: A Performance Study,” *Lecture Notes in Computer Science*, number 1134. Editors: R. R. Wanger and H. Thoma. Springer Verlag, pages 708--717, September 1996.
- [5] E. Pitoura and B. Bharat Bhargava, “Maintaining Consistency of Data in Mobile Distributed Environments,” *The 17th International Conference on Distributed Computing Systems*, pages 404--413, 1995.
- [6] Rifaat A. Dayem, “Mobile Data and Wireless Lan Technologies,” *Prentice Hall*, 1997.
- [7] D. Barbara and T. Imielinski, “Sleepers and workaholics: caching strategies in mobile environments,” In *Proceedings of ACM-SIGMOD 1994 International Conference on Management of Data*, Minneapolis, Minnesota, pages 1--13, May 1994.
- [8] Ahmed Elmagarmid, Jin Jing and Tetsuya Furukawa, “Wireless Client/Server Computing for Personal Information Services and Applications,” *SIGMOD Record*, Vol. 24, No. 4, pages 16--21, December 1995.
- [9] Tomasz Imielinski and B. R. Badrinath, “Wireless Mobile Computing: Challenges in Data Management,” *Communications of the ACM*, 37(10):18--28, 1994.