Wireless Client/Server Computing for Personal Information Services and Applications

Ahmed Elmagarmid Jin Jing Tetsuya Furukawa *
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907 USA
{ake,jing,furukawa}@cs.purdue.edu

Abstract

We are witnessing a profound change in the global information infrastructure that has the potential to fundamentally impact many facets of our life. An important aspect of the evolving infrastructure is the seamless, ubiquitous wireless connectivity which engenders continuous interactions between people and interconnected computers. A challenging area of future ubiquitous wireless computing is the area of providing mobile users with integrated Personal Information Services and Applications (PISA). In this paper, a wireless client/server computing architecture will be discussed for the delivery of PISA. Data management issues such as transactional services and cache consistency will be examined under this architecture.

1 Introduction

Data management in mobile computing environments has recently become an area of increased interest to the database community [5, 1, 2]. In future mobile computing environments, a large number of users carrying low-power palmtop machines will access information over wireless channels anywhere and at any time. An example of future mobile environments is the so called *Personal Communication Services* (PCS) network, though it is still far from being final [9, 6]. The PCS network is expected to be based on current cellular network architecture. The applications and services provided by the future PCS network, called *Personal Information Services and Applications* (PISA), will include personalized financial and stock market information, electronic magazines, travel information, mobile shopping and banking, and electronic mail services [9, 6]. An important and challenging area of mobile computing is the design of architectures and protocols for providing mobile users with PISA.

In the future PCS network environment, mobile users connect to information servers or other users through either wireless or wired connections. In general, the connections can involve the exchange of voice, data, text, facsimile or video information. The PCS network consists of a wired portion and a wireless portion. The wired portion consists of a signaling network which transmits control information for locating users, establishing and tearing down connections, etc., and a separate transport network which transmits the actual user information. The wireless portion of the PCS network transmits both control and user information.

For the delivery of PISA, distributed data servers are connected to mobile users via a PCS network. Mobile users interact with distributed data servers through client processes which run in mobile hosts. These interactions are divided into logical, application-dependent segments called "sessions" which follow a client/server computing paradigm.

However, the wireless client/server paradigm is distinguished from the conventional client/server paradigm by the mobility and portability of the client hosts. First, the mobility of the hosts implies that the hosts will connect from different access points and may also want to stay connected while on the move. Second, for mobile hosts, the primary connection to the rest of the network is a wireless link. Wireless links are relatively unreliable and have low bandwidth constraints. Furthermore, mobile hosts equipped with batteries suffer from limited battery life constraints.

In this paper, we will describe a client/server computing architecture based upon the future PCS wireless network for the delivery of PISA. We will discuss the computing characteristics of this architecture. Then, we will address data management issues such as *transactional services* and *cache consistency* under this architecture. The discussions will emphasize how computing characteristics of wireless client/server computing impact data management issues and what new algorithms are required to deal with these impacts.

The paper is organized as follows. Section 2 discusses a wireless client/server computing model for the delivery of PISA and the computing characteristics of the model. In Section 3, we address the data management issues for the delivery of PISA. The concluding remarks are offered in Section 4.

2 Wireless Client/Server Computing for the Delivery of PISA

2.1 Wireless Client/Server Computing Model

There are different kind of architectures for information servers which can be supported by the underlying PCS networks. The different types of architectures depend on both the information structure supported by the servers and the

*Visiting Associate Professor from Kyushu University, Japan

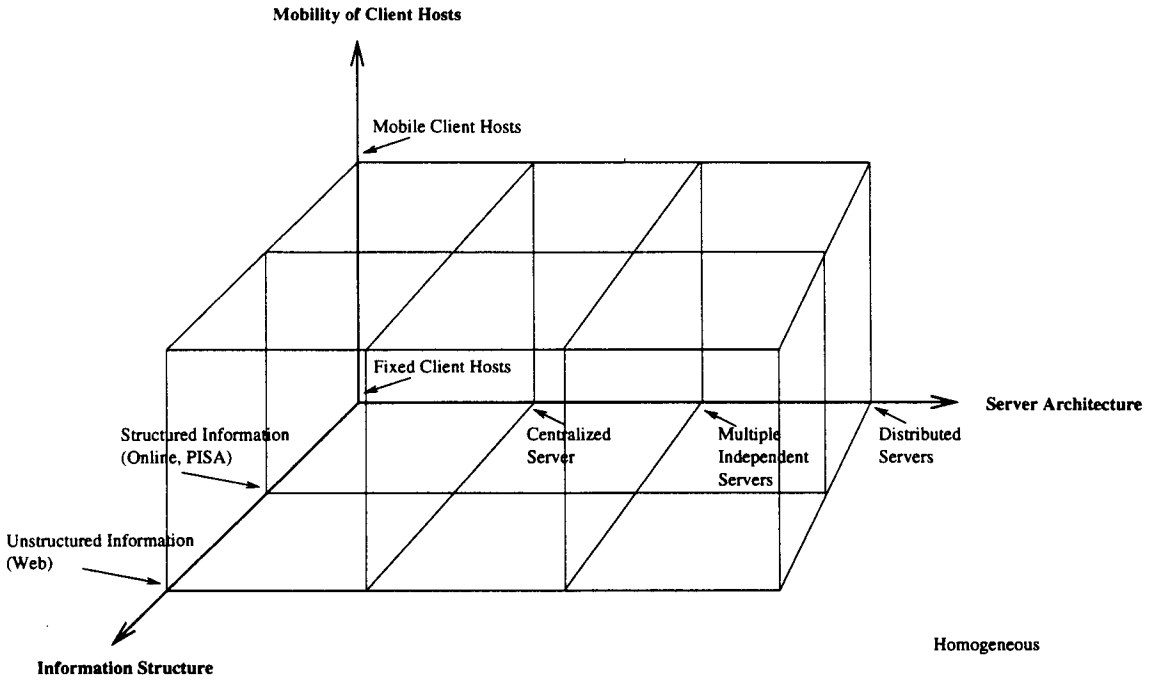

Figure 1: Architecture Alternatives for Information Services

mobility of client hosts. A framework of these alternatives is illustrated in Figure 1. In [6], the architectures of information servers for the delivery of PISA in mobile environments are summarized as the following:

- *centralized architecture*;
- *multiple independent servers architecture*;
- *distributed server architecture.*

In a centralized architecture, a central site stores and processes personalized information for mobile users via a PCS network. If the mobile users are geographically localized and move infrequently, the architecture may be sufficient to support PISA. However, current market and technology trends project rapid increases in the number of mobile users with intelligent mobile computing and communication devices [9, 4]. These users will have increased demand for nationwide or worldwide delivery of PISA. If these trends evolve as projected, the centralized architecture will become inviable, largely because possible computing and communication bottleneck at the central site. Initially, this could be addressed by installing a *centralized parallel server* at the central site, i.e., a logically centralized server which physically consists of several processors working in parallel. However, as the user base becomes more geographically dispersed, the communication costs and delays involved in interacting with users from a central server site will become unacceptable. To address these communication concerns, multiple independent servers at several geographically distributed sites can be installed. Each of these servers can be connected independently to the PCS network.

However, in general, mobile users will want access to private and corporate databases which cannot be geographically allocated into locally accessed portions. It will then be necessary to use a distributed server architecture in which the information is partially replicated or partitioned across multiple interconnected servers with the system functioning as a single logical information base. There are several possible ways of interconnecting the servers, e.g., using a private information service network attached to the PCS network via a gateway or using the PCS network itself as the inter-server communication backbone. For the rest of this paper, we assume a distributed server architecture for the delivery of PISA to mobile users.

Distributed information servers connected to mobile users via a PCS network will contain the appropriate information and provide services for the delivery of PISA. Mobile users interact with distributed data servers through client processes which run in mobile hosts. These interactions are divided into logical, application-dependent segments called "sessions" which use a client/server computing paradigm.

Figure 2 is our proposed architecture for the wireless mobile client/server computing paradigm for the delivery of PISA. A *Mobile Client Host* (MCH) is a computer that can move while retaining its network connections through wireless communication. A *Mobile Support Station* (MSS) is a computer that is connected to a fixed network through wired communication. *Wireless Cell* is a geographical coverage area serviced by an MSS. An MCH can only communicate with an MSS if the MCH is physically located with the cell serviced by the MSS.

Distributed Information Server is the software that runs at an MSS or fixed host and provides mobile application services and information to the mobile hosts. The geographical coverage area for the information service is partitioned into *service areas.* It is likely that a service area will cover several wireless cell. Each service area is served by a signal information server called the *local server. Mobile Client* is the software that runs at an MCH and supports query invoking and information filtering for the delivery of PISA.

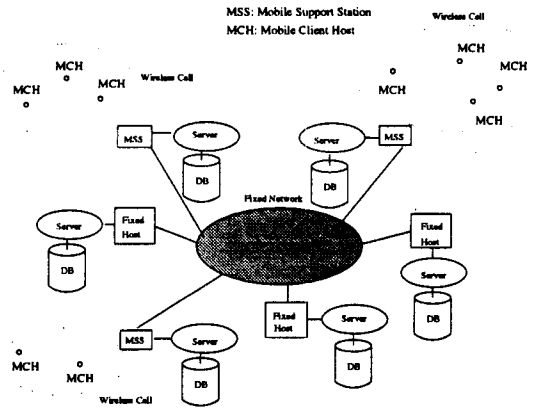

Figure 2: A Wireless Mobile Client/Server Architecture for PISA

The PCS network's most basic function is to support the wireless connection between the mobile hots and the PCS networks. We assume that the protocols for performing the physical connection transfer as the user moves from one cell to another are provided by the underlying PCS network. We also assume that the link-level protocol in the PCS network recovers from bit errors, packet losses, duplications, and reordering for both wireless and wired links.

2.2 Wireless Client/Server Computing Characteristics

The wireless client/server paradigm is distinguished from the conventional client/server paradigm by resource constraints such as wireless bandwidth, battery life and the mobility of hosts. First, the mobility of the hosts implies that they will connect from different access points and may want to stay connected while on the move. Second, for mobile hosts, the primary connection to the rest of the network is a wireless link. Wireless links are relatively unreliable and have low bandwidth constraints. Moreover, mobile hosts powered by batteries suffer from limited battery life constraints.

The physical characteristics in mobile environments pose new computing characteristics for wireless client/server paradigm. These computing characteristics include:

- *asymmetrical communication between clients and servers.* Clients at mobile hosts usually have a weaker capability for transmitting messages than servers at fixed hosts because of resource constraints such as limited battery power. Generally, receiving messages is less costly than sending messages for mobile hosts;
- *long disconnections of clients.* Mobile clients are likely to work in a disconnected or weakly-connected mode to conserve battery consumption.
- *virtual mobility of servers.* Local or nearby data servers can provide more efficient services to the mobile hosts. Therefore, the mobility of the mobile hosts will introduce the virtual mobility of servers which provide clients with information services.

To effectively support PISA, the following data management capabilities in wireless client/server computing environments will be very useful:

- Supporting *long-duration execution* of applications at distributed information servers. This feature allows a mobile application to be issued and monitored by a mobile user from a mobile computer which is often disconnected to save battery power.
- Managing *migration hand-off and recovery* while a mobile user moves between service areas. This feature is required because the movement of mobile hosts may happen during the execution of queries in distributed information servers.
- Supporting *scalability* to large numbers of Mobile Client Hosts. This feature is important for providing services to the massive number of mobile users in the wireless environment with narrow bandwidth channels. The feature will need to deal with how the information is organized in the wireless channels; what information should be disseminated from the channels; and when the information will be downloaded from information data servers to mobile client hosts, etc.

3 Data Management Issues for the Delivery of PISA

We have discussed a client/server architecture in a distributed mobile environment with narrow wireless bandwidth and variable connectivity. In this architecture, information (data) servers are connected via a fixed network to provide information services (e.g. PISA) to mobile users. The information provided by these servers are well structured and replicated (or partitioned) in geographically distributed sites.

Some interesting data management issues arise under the wireless client/server computing architecture. For example, in a repliacted or partitioned database environment, the transaction logs that record both control information (e.g. locks) and updated data values may be stored in different sites as the result of service handoffs for mobile hosts. What kind of protocols can be used to efficiently manage such distributed transaction logs? For well-structured information (or data) services, client cache is a common used technique to improve query response time. How does the frequent disconnection of mobile hosts affect cache invalidation strategies? In this section, we will discuss data management issues including *transactional services* and *cache consistency* under the wireless client/server computing architecture. Our discussions focus on identifying how computing characteristics in wireless client/server computing impact these data management issues and what approaches are required to deal with these impacts.

3.1 Transactional Services and Handoffs

The examples of transactional services in the future wireless mobile environments include electronic commerce applications such as shopping, banking, sales, and inventory, etc.

The transaction submitted by the mobile host is termed a *mobile transaction* and the host is called a *mobile transaction host.* A mobile transaction is a distributed transaction where some parts of the computation are executed on mobile hosts and other parts on fixed hosts. Distributed information servers support basic transaction operations such as read, write, prepare, commit, and abort.

Each MSS has a *coordinator* which receives transaction operations from mobile hosts and monitors their execution in the database servers within the fixed networks. For example, the coordinator will send a read operation to a local server if the copy to be read is in the local site. Similarly,

for a commit operation, the coordinator monitors the execution of 2PC protocol over all the servers involved in the execution of the transaction.

A mobile host may submit transactions in one of two ways:

- An entire transaction may be submitted in a single request message; the whole transaction thus becomes one submission unit. The mobile host also delivers execution control to its coordinator and awaits the return of the results of the transaction execution.
- The operations of a transaction may be submitted in multiple request messages. A submission unit thus consists of one operation (e.g., read) or a group of operations; the mobile host interactively submits the operations of a transaction to its coordinator. A subsequent operation can be submitted only after previous submissions have been executed and the results returned from the coordinator.

While the first approach involves a single coordinator for all of the operations of a transaction, the second approach may involve multiple coordinators because of the mobility of the host. For example, a mobile host may move into a new cell after it obtains the results of previously submitted operations. In the new cell, it will submit the remainder of the transaction operations to the coordinator in the appropriate new MSS. The second approach supports the interactive execution of transactions and, therefore, offers increased flexibility in transaction computations.

3.1.1 Characteristics of Mobile Transactions

It is necessary to reexamine various transaction processing issues in mobile computing systems to account for characteristics of wireless client/server computing. We assume that a mobile host interactively submits the operations of a transaction to its coordinator. The latter subsequentially forwards these operations to the distributed database servers. The characteristics of a mobile transaction can be summarized as follows:

- A mobile transaction tends to be *long-lived* because of high latency of wireless communications and long disconnections of clients.
- A mobile transaction tends to be *error-prone* because mobile hosts are more prone to accidents than fixed hosts.
- A mobile transaction may access a *distributed and heterogeneous* database because of the mobility of the transaction hosts.

Mobile transactions are long-running, error-prone, and heterogeneous. As a consequence, modeling mobile transactions as ACID transactions may be very restrictive.

3.1.2 Data Consistency

A mobile host is only intermittently connected to the rest of the network. Thus, it is crucial to allow a mobile host to operate even while disconnected. Maintaining consistency of data under disconnection conditions is a very difficult task. Some possible techniques to deal with data consistency under mobile environments include:

- read-any/write-any weakly consistent replications. Replication is required in order for disconnected users to access a common database. To maximize a user's ability to read and write data, even while completely disconnected from the rest of the computing environment, the techniques for managing weakly consistent replicated data are required, not only for the high availability but also for the scalability and simplicity.
- deferred convergence control for eventual consistency of replicated data. Servers propagate writes among copies of the database using deferred convergence control algorithms to eventually converge database into identical states if there are no new updates.
- dependency checks on each write. Because clients may make concurrent writes to different servers or may attempt to update some data based on reading an out-of-date copy, update conflicts are unavoidable in a read-any/write-any replication schema. To support detection of update conflicts, dependency checks are necessary.

3.1.3 Service Handoffs

We first briefly discuss how the usual PCS call handoff mechanism can be extended to support virtual mobility of servers for transactional services (A detail discussion about this issue can be found in [6]). We then discuss the difference between the transactional service handoffs and non-transactional service handoffs. In this discussion, *call handoff* for mobile hosts means the physical connection transfer between the old and new support station, while *service handoff* is used to describe the connection transfer for mobile clients between the old and new information servers.

Service handoffs are useful because they help reduce transmission delays caused by mobility. Note that the movement of mobile hosts may result in a long path of communications in fixed networks because the physical distance does not necessarily reflect network distance in mobile environments. In addition, service handoffs may help balance server workloads.

In a PCS network, the mobile client monitors the radio signal strength it receives from neighboring base stations while a call is in progress. Suppose the user moves so that the signal received from a new base station is substantially stronger than that received from the old base station. The client requests a call handoff by signaling to the new base station and then continues communication with the old base station. The new base station initiates the set-up of a three-way bridge, similar to that used for a voice conference call, so that the original connection between the old base station and the party with which the client is communicating is temporarily converted to a three-way call. The client is informed via the old base station once the bridge set-up is complete, and the client switches to the new base station. The client then informs the new base station that it has switched and the original connection between the old base station and the party with which the client is communicating is torn down.

The key procedure about the call handoff above is that it is a temporary call bridge arrangement initiated by a base station. For transactional services handoffs, the physical connection transfer between the old and new server can be done in a similar way. Thus, the transaction coordinator, when informed that the user has moved, can initiate the set-up of a conference call between the current server, the mobile host, and the new server so that the service can be

transparently handed off to the new server. The coordinator can then terminate the call with the old server.

Before the new server takes over during a service handoff, it has to know what the mobile user is currently doing with the service, i.e., the *context* of the user with respect to the service. Context information is the information associated with a user and a service so that the user can access different servers transparently. Part of the context is static, including password and access rights that do not change as the user accesses information. The context also includes dynamic information that indicates session-specific data, such as how much of the data has been read or modified by the user, whether the changes are meant to be transactional, whether the user holds any locks to access the data and so on. In our research work, we only focus on the dynamic context information transfer among distributed data servers.

We note that a service handoff need not occur immediately after the user moves from one service area to another. Thus, the coordinator can suggest to the old server that a service handoff should occur, but the handoff can be delayed by the old server until a logical handoff initiation point is reached in the information transfer to the mobile client. For instance, suppose a user is reading a news magazine and the information is being transmitted page by page. If a particular page is being read while the user moves from one service area to another, the service handoff can be delayed until after the entire page has been transmitted. The ability to delay the initiation of service handoffs after a user changes service areas allows the old server to choose the most logical and convenient point for performing each handoff and to determine the context information which must be transferred for each class of applications.

A question arised from transactional service handoffs is how transaction logs could be managed efficiently. For non-transactional service handoffs (including call handoffs) processing, the context need only include the current execution status of applications. In contrast, in addition to the current execution status information, the history information of the transaction executions that is recorded in the transaction logs should also be taken into account for the service handoff processing.

There are different solutions for the management of distributed transaction logs. One solution does not require the handoff protocols to transfer the log information from the old server to the new server. At the commit time, however, a global commitment protocol is used to restore log information back into databases. The second solution involves the log transfers during the transaction service handoffs and local commitment of transactions at the commit time. Note that we assume that all data are partitioned or replicated among the distributed database servers so that the same database services could be supported in the distributed environments.

In [7, 3], we illustrate that, in some cases, neither transaction log transfers nor global commitment is needed for the transactional service handoff processing. Specifically, we proposed a new approach that utilizes the commutativity semantics of operations on partitioned or replicated data to optimize message traffic among the distributed data servers for transactional service handoffs. The key idea behind this approach is that the log information related to commutative operations can be left in different servers until non-commutative operations on the same data from other transactions are executed. For example, in a replicated database, the read lock and unlock for a data item in a transaction can be recorded in different copy sites (note that read locks and unlocks of two transactions are commutative, assuming a read-one-write-all locking protocol is used). The read lock and unlock logs of a transaction can be left in different sites even after the commitment of the transaction until a write lock is requested. Remote messages to read lock sites are unnecessary at the commit time of a transaction that issues these read locks. Therefore, the deferred log updates for commutative operations can be utilized to reduce message cost for distributed log managment for mobile transactions.

A prototype of the transactional service for mobile hosts is currently being implemented on the Coda file system [10, 12] to support continued services in a disconnection mode. The service handoff issue with the distributed data server architecture have been discussed [6, 11, 7, 3].

3.2 Cache Consistency

In a wireless client/server computing environment, caching of frequently-accessed data items is an important technique that will reduce contention on the narrow bandwidth wireless channels. The cached data can help to improve query response time and to support disconnected or weakly connected operations. However, cache consistency strategies will be severely hampered by both the disconnection and mobility of clients since a server may be unaware of the current locations and connection status of clients. Existing caching consistency algorithms in traditional client/server architecture in which the location and connection of clients are not changed can be divided into two categories:

- *callback approach*: servers send invalidation messages directly to clients that have cached the data items to be updated;
- *detection approach*: clients send queries to servers to validate cached data.

Since mobile client hosts often disconnect to conserve battery power and are frequently on the move, the callback approach is not easily implemented in the mobile environment. On the other hand, the narrow bandwidth wireless network would be clogged if massive numbers of clients attempt to query a server to validate cached data. Also, in a wireless environment, mobile clients usually have weak or little transmission capability because of limited battery power while stationary servers have powerful broadcast transmitters. Therefore, both callback and detection approach employed in the traditional client-server architecture are not be readily applicable to mobile environments.

In [2], Barbara and Imielinski proposed an approach to the problem of invalidating caches in mobile environments. In this approach, a server periodically broadcasts an *invalidation report* that reports the data items which have been changed. Rather than querying a server directly regarding the validation of cached copies, clients listen to these invalidation reports over wireless channels. This approach is attractive in the mobile environment because:

- a server need not know the location and connection status of its clients. The servers with the *stateless* characteristic will be able to adapt well to the mobility and disconnection of clients;
- the clients need not establish an "uplink" connection to a server to invalidate their caches. The *weak connection* mode will effectively support mobile clients with limited battery capability and in narrow bandwidth wireless channels.

One interesting new issue, however, arises when clients use the invalidation report to invalidate their caches. This is the issue of *false invalidation* of client caches. Since an invalidation report can include only limited or partial information regarding data changes or the current database state, caches that are actually valid may be invalidated. For example, when a client has disconnected for a substantial period of time, the report may not cover all updates to data items that have occurred since the time of disconnection. In this case, the client may invalidate a cache which is actually valid. Once a cache is invalidated, the client must seek verification or updating of the cache from the data server.

In a mobile environment, it is desirable to provide a cache invalidation algorithm whose performance will not be severely affected by changing workload parameters such as client disconnection times and update rates. Mobile units will frequently be "off," with substantial periods of disconnection intended for battery conservation. Within these substantial disconnection periods, the number of items to be updated by a server may be unpredicatable. A high rate of false invalidation will usually reduce the cache hit rate in query processing, resulting in increased traffic on the narrow wireless networks and decreased query processing throughput.

It is very useful to investigate new invalidation-report based cache consistency algorithms with the following features:

- the *effectiveness* of algorithms should be less adversely impacted by changing workload parameters, such as client disconnection times and update rates/patterns etc;
- these algorithms should *scale* to large databases without the use of large invalidation report.

The scalability feature is important because the large invalidation report will waste precious wireless bandwidth and increase latency for client cache verification. Specifically, these scalable algorithms should be well suited for applications such as "information feed" application domains.

These existing algorithms in [2] are *effective* for a client in terms of a low ratio of false invalidation (to total invalidation), only if the client has not disconnected for a period that exceeds an algorithm-specified parameter or if the number of updated items during the period is not greater than an algorithm-specified parameter (see [8] for detail). The scalability issue has been addressed in [8].

4 Summary

The rapidly expanding technologies of wireless communications and personal computing introduces an entirely new computing environment called *mobile computing*. The concept, approaches, and methodology in traditional information systems need to be reexamined in order to fit the new computing environment. We have described a wireless client/server computing architecture for the delivery of *Personal Information Services and Applications* (PISA) and the computing characteristics of the architecture. These characteristics include asymmetrical communication between clients and servers; long disconnection of clients; and virtual mobility of servers. We have also discussed such data management issues as transactional services and cache consistency under the wireless client/server paradigm for the delivery of PISA.

Many issues and problems raised here still remain to be solved. We are currently investigating some of the issues raised in this paper, including transactional service handoff protocols and cache invalidation algorithms through the use of broadcast channels. Specifically, we are interested in the protocols and algorithms that are well suited to the personal information services and applications in future mobile wireless environments.

References

[1] R. Alonso and H. Korth. Database issues in nomadic computing. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 388–392, 1993.

[2] D. Barbará and T. Imielinski. Sleepers and workaholics: Caching strategies for mobile environments. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 1–12, 1994.

[3] A. K. Elmagarmid, J. Jing, and O. Bukhres. An efficient and reliable reservation algorithm for mobile transactions. Technical Report CSD-TR-95-018, Department of Computer Science, Purdue University, 1995. Appeared in shorter form in Proceedings of the 4th International Conference on Information and Knowledge Management (CIKM'95).

[4] G. Forman and J. Zahorjan. The challenges of mobile computing. *IEEE Computer*, 27:38–47, 1994.

[5] T. Imielinski and B. R. Badrinath. Wireless mobile computing : Challenges in data management. *Communication of ACM*, 37(10), 1994.

[6] R. Jain and N. Krishnakumar. Service handoffs and virtual mobility for delivery of personal information services to mobile users. Technical Report TM-24696, Bellcore, Dec. 1994.

[7] J. Jing, O. Bukhres, and A. Elmagarmid. Distributed lock management for mobile transactions. In *Proc. of the 15th International Conference on Distributed Computing Systems*, Vancouver, Canada, June 1995.

[8] J. Jing, O. Bukhres, A. Elmagarmid, and R. Alonso. Bit-sequences: A cache invalidation algorithm in mobile environments. Technical Report CSD-TR-94-074, Department of Computer Science, Purdue University, 1994.

[9] R. H. Katz. Adaptation and mobility in wireless information systems. *IEEE Personal Communications*, 1:6–17, 1994.

[10] J. Kistler and M. Satyanaranyanan. Disconnected Operation in the Coda File System. *ACM Transactions on Computer Systems*, 10(1), February 1992.

[11] N. Krishnakumar and R. Jain. Protocols for maintaining inventory databases and user service profiles in mobile sales applications. In *Proceedings of the Mobidata Workshop*, Rutgers University, Nov. 1994.

[12] Q. Lu and M. Satyanaranyanan. Isolation-only transactions for mobile computing. *ACM Operating Systems Review*, 28(3), July 1994.