Building Information Systems for Mobile Environments*

Evaggelia Pitoura
Department of Computer Science
Purdue University
West Lafayette, IN 47907-1398
pitoura@cs.purdue.edu

Bharat Bhargava
Department of Computer Science
Purdue University
West Lafayette, IN 47907-1398
bb@cs.purdue.edu

Abstract

It is expected that in the near future, tens of millions of users will have access to distributed information systems through wireless connections. The technical characteristics of the wireless medium and the resulting mobility of both data resources and data consumers raise new challenging questions regarding the development of information systems appropriate for mobile environments. In this paper, we report on the development of such a system. First, we describe the general architecture of the information system and the main considerations of our design. Then, based on these considerations, we present our system support for maintaining the consistency of replicated data and for providing transaction schemas that account for the frequent but predictable disconnections, the mobility, and the vulnerability of the wireless environment.

Keywords: New Applications, Information Systems, Mobile Computing, Transaction Management, Consistency.

1 Introduction

In the recent past, technical advances in the development of portable computers and the rapidly expanding cordless technology have provided the basis for accessing information systems through wireless connections. Today, when users move, unplug their computer from some local area network, transport it, and plug it back to the local area network at their destination. Wireless technology provides users with the ability to retain their network connection even while moving. This new computing paradigm is called *mobile* or *nomadic* computing. Mobile computing can be viewed as adding a new dimension to the broader vision of universal access to information that allows the mobility of data consumers and data resources.

*In Proceedings of the 3rd International Conference on Information and Knowledge Management, Gaithesburg, MD, Nov 1994, pp 371-378
This and related reports are available through www, URL address: http://www.cs.purdue.edu/people/pitoura

Until recently, infrastructure research pertaining to mobile computing has mostly focused on networking and operating systems [18, 8, 17, 25, 4, 23]. Research in networking and communications includes new addressing and routing schemas, support for efficient multicasting and broadcasting, data compression, and relocation transparency. Research in operating system addresses security issues, file systems that support disconnected operation, and caching techniques. However, the issues introduced go beyond those areas and directly affect information management systems [3, 15, 20]. Mobile computing introduces new challenging problems concerning resource management, information acquisition [16], and data distribution [13].

In this paper, we report on the design of an information system for a mobile environment. The goal of this paper is twofold. First, we give a general overview of the organization of the system and of the important concerns of our design. Second, we focus on our system support for consistency and transactions and show how our schema is in compliance with the general architecture and design concerns.

The structure of this paper is as follows. In Section 2, we introduce the physical architecture, and identify the characteristics of both the wireless medium and the mobile hosts. In Section 3, we define the operation modes of a mobile host, present the general object-based architecture of the information system, and introduce the main considerations of the design. The following two sections focus on consistency and transaction management. Section 4 describes our schema for maintaining the consistency of replicated data. The novelty of this schema is the existence of two types of operations (loose and strict) that allows users to specify the required degree of consistency of their input data. The schema takes into consideration the modes of operation and the peculiarities of the mobile environment. Section 5 discusses the structure of a mobile computation. We argue that flat transactions are inadequate for modeling interactions in a mobile environment and we propose appropriate extensions that account for mobility and recovery. In Section 6, we report briefly on the status of our system. We conclude in Section 7 by summarizing the main results of our work.

2 The Characteristics of the Mobile Environment

Distributed systems that support mobility are physically structured as shown in Figure 1 [15]. The architecture consists of two distinct types of hosts: mobile and fixed hosts. Some of the fixed hosts, called *base stations* or *mobile support stations*, are augmented with a wireless interface to communicate with mobile hosts. The geographical area covered by a base station is called a *cell.* Each mobile host can directly communicate with one base station, the one covering the geographical area in which the mobile host moves. The process during which a mobile host enters a new cell is called *hand-off.* To accommodate smooth hand-off, cells usually overlap.

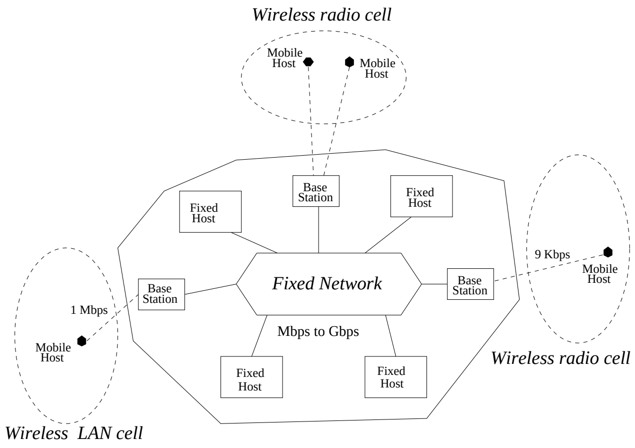

Figure 1: Mobile System Architecture

2.1 The Wireless Medium

The networking infrastructure (also called Personal Communication Network (PCN)) is expected to include [9, 15, 3]:

- *cellular or packet radio* modems (e.g., Ericson GE's Mobidem), which are characterized by high costs, large range coverage, and relatively small bandwidth;
- *satellite services* (e.g., Motorola's Iridium), which provide wide coverage, but are very expensive, usually receive-only and of very low bandwidth; and
- *wireless LANs* (traditional LANs extended with a wireless interface, e.g., NCR WaveLAN, Motorola's ALTAIR, Proxims Range LAN and Telesystem's ARLAN), which provide connectivity with low cost within a very small geographical area (at a range of few kilometers).

While the growth in physical network bandwidth has been tremendous (in current technology Ethernet provides 10Mbps, FDDI 100 Mbps and ATM 155 Mbps), products for wireless communication achieve only 2Mbps for radio communication, and 9-14 kbps for cellular telephony [9]. The typical bandwidth of wireless LANs ranges from 250 bps to 2Mbps and it is expected to increase to 10Mbps. Since the bandwidth is divided between the users sharing a shell, the deliverable bandwidth per user will be even lower. Thus, it is safe to assume that bandwidth will continue to be a scarce resource. Furthermore, an additional reason that makes bandwidth consumption a major concern of mobile computing designs, is that data transmission over the air is *monetarily expensive* [12].

In comparison with non-mobile environments, disconnections are much more frequent. Furthermore, the wireless medium is characterized by much greater variation in network bandwidth than traditional designs, leading to various *degrees of disconnections* depending on the available bandwidth and noise of the communication channel [9, 14]. Certain disconnections are considered *foreseeable,* since they can be detected by changes in the signal strength, by predicting the battery's lifetime, or by utilizing knowledge of the bandwidth distribution [3, 14].

Product	RAM	MHz	CPU	Batteries hours, type	Weight lbs	Display sq. inches
Amstard Pen Pad PDA600	128 KB	20	Z80	40, AAs	0.9	10.4
Apple Newton Message Pad	640KB	20	ARM	6-8, 4 AAAs	0.9	11.2
At&T E0 440	4-12 MB	20	Hobbit	1-6 , NiCad	2.2	25.7

Table 1: Typical Values of Mobile Hosts

2.2 Mobile Hosts

Mobile hosts, regardless of future technology advances, will have limited computing power, memory and screen size as a result of their small size and weight. Some typical values are shown in Table 1 [9]. Mobile hosts range from small palmtops (e.g., Apple Newton) or specialized data entry devices (like the ones used by UPS) to tabletop computers with wireless interfaces (e.g., AT&T EO). Mobile hosts have limited battery capacity, two or three hours under normal use, which is expected to increase only 20% over the next 10 years. Due to this fact, energy preservation is an important consideration. Moreover, mobile hosts are more susceptible to loss, destruction and theft than static hosts. Table 2 summarizes the characteristics of the wireless medium and of the mobile hosts.

characteristics of the wireless medium	**characteristics of mobile hosts**
low bandwidth frequent disconnections high bandwidth variability predictable disconnections monetarily expensive broadcast is physically supported in a cell	small size small screen limited battery life susceptible to theft, and accidents

Table 2: Summary of the Characteristics

3 System Overview

In this section we provide an overview of an information system for a mobile environment. First, we define the modes of operation, then we describe the general architecture, and finally, we conclude with two important considerations in the design of the system.

3.1 Operation Modes

While in a nonmobile distributed system a host operates in one of two modes regarding its connection to the rest of the network (either connected to it or totally disconnected from it) in a mobile environment there are more modes of operation. We model the different modes of operation of a mobile host using the state diagram shown in Figure 2.

The degree of connection is related to the availability of bandwidth. Since total and partial disconnections are very frequent, they should not be treated as failures. On the contrary, a mobile host should be capable of operating even under low or no connection with the fixed network. A mobile host operates in a *partly disconnected* mode when the communication through the wireless network is weak.

In addition to the modes related to the type of connection with the fixed network, a mobile host may switch to the *doze mode* for preserving energy. While in this mode, the clock speed is reduced and no user computation is performed. The mobile host returns to normal operation upon receipt of any message.

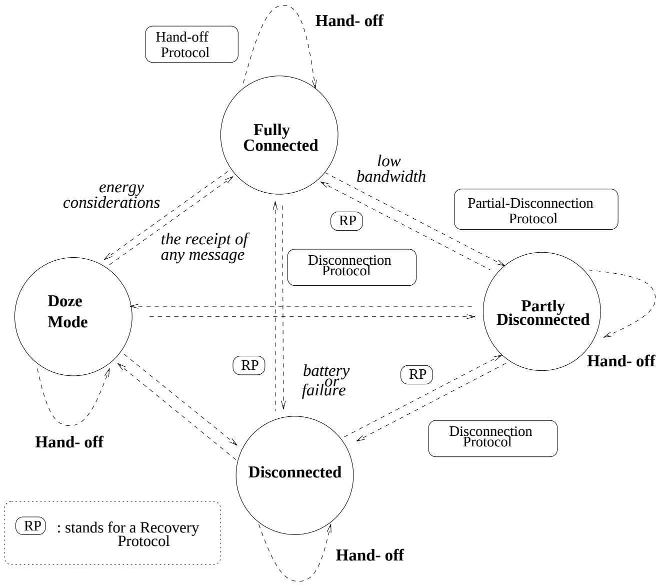

Figure 2: Operation Modes of a Mobile Host

Since most of the transitions between modes are predictable, protocols can be designed to prepare the system for a transition.

- A *disconnection protocol* is executed before the mobile host is physically detached from the fixed network. The protocol should ensure that enough information is locally available to the mobile host for its autonomous operation during disconnection. It should also notify any interested parties for the forecoming disconnection.
- A *partial-disconnection protocol* prepares the mobile host for operation in a mode where all communication with the fixed network should be as restricted as possible. Selective caching can be used for caching data, whose presence in the host will minimize future network use.
- *Recovery protocols* re-establish the connection with the fixed network and resume normal operation.
- *Hand-off protocols* refer to crossing the boundaries of a cell. State information pertaining to the mobile host should be transferred to the base station of the new cell.

Part of these protocols is handled by low-level system operations. However, the modes are not necessarily completely transparent to the applications. Support by the application (in our case by the information management system) can lead to performance gains. For example, information about the type of application can be used to design caching techniques or limit network access.

In this paper, we focus on how system support for transactions and consistency maintenance can take into account the modes of operation to increase performance and availability. Specifically, in Section 4, we present a replication controller, which adjusts with the type of connection. In Section 5, we introduce the concept of transaction migration to handle hand-offs.

3.2 The Architecture of the Information System

Users of mobile hosts will have access to a variety of information resources, which will be located in both mobile and static hosts. An architecture for such systems must take into account the following facts:

- Although, some of the information resources will be provided by information systems especially tailored for such use, many of them will be provided by *pre-existing* applications that were built without taking into account the possibility of their future access through wireless connections. Rather than rebuilding the applications, superimposing upon them appropriate interfaces is a more realistic approach.
- An intrinsic characteristic of these information resources is their *heterogeneity*. Heterogeneity is the inevitable result of an expected increase in the scale of distributed systems with the introduction of mobile hosts. Furthermore, the use of wireless connections results in users entering new cells and possibly heterogeneous environment during a single session.
- The architecture must support extensibility and portability. Compliance with evolving standards such as CORBA [11] and ODP [24] is a step towards this direction.

Because of the above considerations, an *object-based architecture* seems to be appropriate to serve as the architecture of a mobile information system. A Distributed Object Management System (DOMS) consists of a number of distributed hosts and clients [19]. Each host supports one or more information systems. Together, the hosts constitute the system's information resources. Special *Object Servers* or *Object Managers* are responsible for building appropriate interfaces so that the system's resources appear as objects. Clients request operations by sending messages to objects. These messages are handled by object servers which direct them to the appropriate hosts.

Object-based architectures are suitable for the following reasons:

- Special *Mobile-Object Servers* can be built that will offer appropriate methods for accessing data from mobile hosts. These methods will account for the wireless of the medium and the fact that the location of a client is a frequently changing data, see Figure 3.
- Object-based architectures hide the heterogeneity of the environment, since the response to a message depends only on the interface and not on the internal implementation of the object or the method.

- Most proposals for standards are based on the distributed object paradigm.

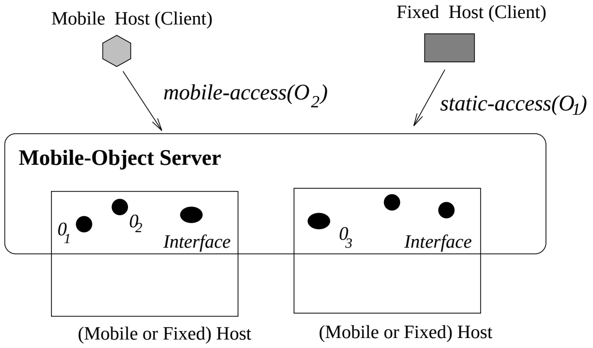

Figure 3: Object-based Architecture, objects O_i correspond to resources stored in static and mobile hosts

The distributed object-based architecture provides the high-level architecture of the distributed heterogeneous information system. Internally, each information system that participates in the object-based architecture may support its own data model. What would be an appropriate internal data model for a local mobile database is not clear yet, but it should support graphical interfaces and take into account the inherent heterogeneity of the mobile environment [2].

Location Databases. In addition to the servers that will provide wireless access to general information resources, the system should maintain servers that offer access to information systems that contain location-related information. We assume at least two such databases, one per user that would include the user's home address and other user-related information and one per base station that would include information about all hosts under its cell. Location databases are:

- fast changing,
- geographical,
- distributed and replicated over many sites to support efficient access,
- imprecise, since the overhead for keeping them up-to-date may be overwhelming.

Data stored in location databases may participate in queries. This results in queries that are:

- of different complexity in terms of location constraints,
- geographical and moreover dependent on the location of the user (e.g. find the nearest restaurant),
- real-time,
- imprecise,
- dependent on the user's direction of move

Query issues involving location data are discussed in [16]. In Section 4, we discuss issues regarding the consistency of such data.

3.3 Two Important Considerations

Two issues of particular interest in designing information systems for mobile environments are to determine the role of a mobile host in a distributed computation and to *personalize* the computation. We discuss them further.

The Role of a Mobile Host. There is no agreement yet on the role of a mobile host in a distributed environment. The possibilities range from that of a dumb terminal with no autonomy to that of a relatively independent component with enough memory and computing power to perform part of the distributed computation locally. Each approach has both advantages and disadvantages. Performing part of the computation on a mobile host:

- costs in terms of power consumption,
- complicates data replication and integrity preservation,
- adds to the search cost of locating the mobile hosts and the data that they produce and consume.

On the other hand, it

- allows the autonomous operation of a mobile host during partial and total disconnections,
- limits bandwidth use,
- saves the energy cost for the transmission of data to and from the fixed network.

In our design, we adopt the approach that part of the computation will be executed in a mobile host, since allowing a mobile host to operate autonomously is crucial considering the frequency of disconnections. But still, since mobile hosts are susceptible to failures and accidents, part of the computation must be reported to the fixed network. In Section 5, we introduce the notion of transaction proxies that provides transaction support for backing-up the computation in the fixed network.

User's Profile. Utilizing information about users gains increasing importance as information systems scale in size. We assume that this information is stored in a *user's profile*. Information about users is very important to data navigation since it can be used to select the information that is of interest to a particular user. For instance, users may be informed for the release of the latest album of their favored artist. Specifically in mobile environments, the user's profile may also include information about the user's *mobility pattern*, for instance the fact that each day he commutes to his office. The user's profile can be part of the location database of the user.

In mobile systems research the user's profile has been widely used, for example to determine which data to cache before disconnections [18], or to optimize queries about location [16]. In our schema, we propose utilizing information stored in a user's profile to define consistency clusters (Section 4) and to simulate mobility (Section 6).

4 Consistency of Replicated Data

Communication through the wireless network is very expensive both monetarily and in terms of bandwidth consumption. As a consequence, performing operations locally in a mobile host can lead to performance gains and increase availability during total and partial disconnections. On the other hand, maintaining full consistency among data saved in fixed and mobile hosts imposes unbearable overheads in mobile environments [3, 14, 20]. We propose a more flexible model. The formal definition of the model, its scope and the associated correctness criteria are presented in [21]. The presentation of the model in this section aims at demonstrating how the operation modes and the other general considerations are taken into account in designing the replication controller of our information system.

4.1 Clustering

We partition the data items of a mobile database into *clusters*. All replicas in a cluster are synchronized, while replicas at different clusters may vary by some appropriately defined degree. In that sense, a cluster is the unit of consistency. The degree of inconsistency may vary based on the availability of network bandwidth by allowing small deviation among the copies in cases of higher bandwidth availability and higher deviation in cases of low bandwidth availability. The cluster configuration is *dynamic* rather than static. When clusters are merged the values of all copies of a data item are reconciled.

Defining Degrees. There are many alternative ways of defining degrees [22]. The *degree* may express the divergence from the value of the primary copy [1]. In this case, the allowable degree may be bounded by limiting the number of versions, by setting a maximum value on the allowable deviation, or by limiting the number of transactions that can operate on inconsistent values. Alternatively, we can define the degree as the number of data items or the number of data copies per data item that can diverge.

Defining Clusters. One possible way of defining a cluster is by grouping together data residing in the same or neighboring hosts. For instance, data stored in hosts that are partially or totally disconnected from the fixed network may be considered as forming a cluster (Figure 4). In this case, by taking advantage of the predictable nature of disconnections, clusters of data may be created or merged upon a forecoming disconnection or connection of the associated mobile host. Furthermore, clusters may be defined based on the type of data. A characteristic example of such data is the data representing the *location* of mobile hosts. Since this type of data are fast-changing, the maintenance of their full consistency could cause unbearable overheads. In addition, users may explicitly define clusters based on the semantics of their data or applications. Finally, information stored in a *user's profile* may also be utilized to determine clusters. For example, data that are most frequently accessed by a user or data that are to a great extent private to a user can be considered as belonging to the same cluster independently of their location.

4.2 Handling Loose Consistency

To maximize local processing and limit network accesses, we allow users to interact with locally (within a cluster) available data by introducing two new kinds of operations, loose reads and loose writes. These operations allow users to operate on loosely consistent data when the lack of strict consistency does not affect the semantics of their transactions. Allowing users to operate on local data is especially

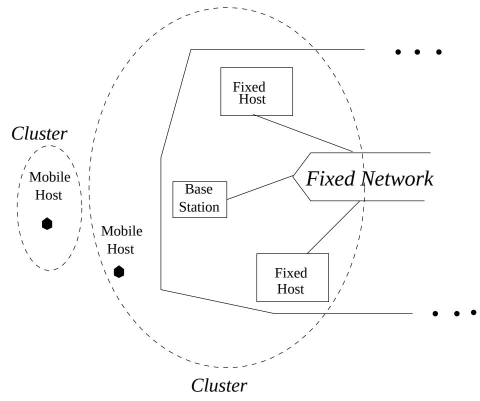

Figure 4: A possible cluster configuration, where clustering is based on location

important during total or partial disconnection since those data are then the only data that are accessible or affordable to access. We call the standard read and write operations strict read and strict write operations to differentiate them from the loose operations.

Specifically, a loose read ($LR[a]$) reads the value written by the last loose or strict write operation on a *locally* available copy, that is on a copy in its cluster. A *loose write* operation ($LW[a]$) writes some local copies and is not permanent unless it is finally committed after cluster merging. A *strict read* operation ($SR[a]$) reads the value written by the last strict write operation on a copy located in *any* of the clusters. Finally, the value written by a *strict write* operation ($SW[a]$) becomes permanent after commitment. Our method can be implemented as a 2-version method, where local transaction managers maintain two copies of a data item, one that is updated by strict writes and one that is updated by both strict and loose writes. The first is read by strict read operations and the second by loose read operations.

Two Types of Basic Transactions. We distinguish two types of transactions:

(a) transactions that consist only of loose read and loose write operations and are called *loose transactions*; and

(b) transactions that consist only of strict read and strict write operations and are called *strict transactions*.

Loose transactions are locally committed in their associated clusters. Updates performed by locally committed transactions are revealed only to other loose transactions in the same cluster. These changes are revealed to strict transactions only after merging, when local transactions become globally committed. Before being globally committed, a loose transaction may be undone even after it has been locally committed.

A loose transaction is a transaction that may read loosely consistent data and whose writes may be undone any time before merging. Strict transactions have the usual semantics. Loose transactions are suitable for applications that do not require exact values of data such as gathering information for statistical purposes. Allowing updates in loose

	$LR_j(a_k)$	$SR_j(a_k)$	$LW_j(a_k)$	$SW_j(a_k)$
$LR_i(a_k)$			X	X
$SR_i(a_k)$				X
$LW_i(a_k)$	X		X	X
$SW_i(a_k)$	X	X	X	X

Table 3: Conflict relation, a "x" entry indicates that the operations for the given row and column conflict. Row entries correspond to operations of transaction T_i and column entries to operations of a transaction T_j, where $i \neq j$

transactions adds functionality. These updates are finally committed only if they do not conflict with the operations of a strict transaction. Thus, loose updates are especially useful in two cases: (a) for handling private or seldomly accessed data for which conflicts are rare, and (b) for transactions for which compensating actions are possible.

Users or application developers are able to specify the loose or strict requirements of their transactions through a high-level interface. Then, the operations of each transaction are automatically translated into loose or strict operations according to the given specifications.

Correctness. Two operations of the same transaction conflict if they both access the same data copy and at least one of them is a write. Operations of two different transactions that are executed before cluster merging conflict as shown in Table 3. Upon merging of clusters we adopt a strictly syntactic approach to establishing full consistency. We undo all loose transactions whose loose writes conflict with a strict transaction. We have proven that undoing a loose transaction can result in undoing only loose transactions in the same cluster [21]. Serializability-based criteria for the correctness of schedules before and after merging are formally defined in [21]. In [21], we have also presented graph-based tests for the serializability of the corresponding schedules and compare our work with other weak consistency proposals in the database and operating system communities.

5 Transactions in a Mobile Environment

Loose and strict transactions are generic transactions that can be considered *part* of a general mobile transaction that models the interaction of a user with a mobile distributed system. Generally speaking, a *mobile transaction* is a distributed transaction T, where some parts of the computation are executed on mobile and some parts on nonmobile hosts. The model for such a transaction that involves data stored in both the fixed and the wireless network has not emerged yet.

A large number of transactions in a mobile environment is expected to be read-only transactions, where users will query large amount of data. Still, some applications such as inventory control, banking applications and travel reservations will require updates. For instance, a traveling salesman will update an inventory database to reflect the fact that an item has been sold. Consumers will use their mobile host to book flights, buy tickets or do banking transactions.

An important issue is what part and how large a part of a transaction will be executed on a mobile host. This is related to the role of a mobile host in a mobile distributed environment. Operations on a mobile host may minimize network access, and optimize response time. On the other hand, they cost in terms of local resource consumption (especially battery). But the decisive factor for answering this question is frequent disconnections. To allow the operation of a mobile host during disconnections part of the computation must be executed locally.

The Characteristics of Mobile Transactions. We identify the following as the characteristics of a mobile transaction:

- Mobile transactions are transactions that involve the wireless network and may be executed in both mobile and nonmobile hosts.
- Using the *wireless medium* has the following consequences. Transactions tend to be:
  1. monetarily expensive;
  2. long lived, because of long network delays;
  3. error-prone, because of frequent disconnections but also because mobile hosts are more prone to accidents than fixed hosts; and
  4. session-based. For some technologies, such as cellular modems, there is a high start-up charge for each communication. Cost-effective transaction management may adopt the approach of supporting few long-lived session-based transactions instead of many short-lived transactions.
- *Mobility* results in transactions with the following characteristics:
  1. Transactions access heterogeneous information systems.
  2. Transactions access (possibly imprecise) location data.
  3. Transactions may involve data that are dynamically relocated.

Modeling Mobile Transactions. Mobile transactions are long-running, error-prone and heterogeneous. As a consequence, modeling mobile transactions as ACID transactions is very restrictive. ACID transactions have limited expressive power and offer no way of modeling computations with a complex control structure. Furthermore, ACID transactions do not support partial commitment or abortion of a transaction, or partial recovery. Finally, there is no way of "suspending" a transaction to survive a disconnection.

It seems that an *open-nested model* [10] is more appropriate for modeling mobile transactions. In that model, each transaction consists of a number of subtransactions with a specified set of dependencies. The set of dependencies varies based on the application and it can be customized. In [7], an axiomatic definition of a transaction model for mobile transactions is presented. Our approach is different in that, instead of defining a powerful, general transaction model, we identify the *generic characteristics* that this model must support to be appropriate for a mobile environment.

Specifically, to deal with the particularities of the wireless medium, we have introduced *loose transactions*. Loose transactions support disconnected operation and minimize bandwidth use. Moreover, they are capable of modeling operations on imprecise data, such as location data. In this section we investigate further on the structure of mobile transactions. We present two generic techniques:

- To deal with the mobility of the environment we introduce the concept of *transaction migration*.
- To deal with the vulnerability of the mobile hosts we introduce the concept of a *transaction proxy*.

5.1 Transaction Migration

A mobile host may enter a new cell while in operation. In that case, it may be necessary to *migrate* part of the computation that was executed in one fixed host to another fixed host. One motivation for migration is improvement in performance. By moving the computation close to the mobile host, the communication cost is minimized. Transaction migration (relocation) can be thought of as the dual of data relocation. In addition, transaction migration may lead to a better load balance among base stations. Furthermore, a base station may not be willing to support computation initiated by users that are not any more in its cell, for instance for security reasons. In that case, transactions submitted to the old base station and partially executed there must be transferred to the new base station.

Let T_i be a transaction that was initially submitted at site i and then was migrated to site j. We use the notation $T_{i \to i}$ for the part executed at site i and $T_{i \to j}$ for the part executed at site j. $T_{i \to i}$ cannot be committed at site i but it may conditionally release some of the local resources it holds. $T_{i \to j}$ inherits state information from $T_{i \to i}$. That information depends on the consistency control method used. It may include timestamps, requested and granted locks, or log files.

5.2 Transaction Proxies

To deal with the fact that mobile hosts are more susceptible to theft accidents and frequent disconnections, we must report part of the computation that is performed in a mobile host to the fixed network. We model that using the concept of *transaction proxies*. For each transaction T_i executed at a mobile host i, we define a dual transaction PT_j, called proxy, that will be executed on host j, where j is the base station of i.

A proxy transaction may be considered as a subtransaction of the original transaction. Thus, any time a subtransaction is submitted to a mobile host its proxy transaction is submitted to its base station. The proxy transaction includes only the updates of the original transaction. Alternatively, proxy transactions may be executed off-line. In that sense, proxy transactions correspond to taking periodic back-ups of the computation which is performed on a mobile host. We are investigating necessary criteria for scheduling proxies at fixed hosts in a sound and efficient way.

6 Status Report

We are currently developing an information system on top of ORAID [5]. ORAID is an object-oriented system built on top of RAID [6], a distributed database system that has been proven successful in supporting experiments in communication, adaptability, and transaction processing. We are modifying the communication library routines [26] to simulate the wireless communication. Mobility is captured by connecting our simulated mobile station with different base stations. We assign to each host that simulates a mobile host, a fixed host which serves as its base station. The mobile host can then directly communicate only with its assigned base station. The assignment of base stations to mobile hosts is dynamic and is driven by the mobility pattern as specified in each user's profile.

In addition to the traditional performance measurements mobile computing introduces new performance concerns, including:

- energy preservation (battery is a limited resource);
- prudent use of bandwidth (bandwidth in mobile environment is a very scarce and expensive resource); and
- the degree of autonomy during weak or total disconnections (since disconnections are frequent, mobile computers should be able to operate even while disconnected).

We intend to provide system support for loose transactions, migration and proxies and study the performance of these mechanisms.

7 Summary

Wireless communications offer the exciting possibility of accessing information independent of location and movement. This possibility raises new challenges to the design of information systems. In this paper we have reported on the design of such a system. The contribution of this paper is twofold. First we have:

- defined the operation modes of a mobile host;
- proposed a general object-based architecture for information systems appropriate for mobile environments; and
- identified the main concerns in designing mobile information systems.

Second, we have shown how these general principles can be used to provide system support for transactions in a mobile environment. Specifically, we have introduced:

- loose transactions to deal with the frequent, predictable and of various degrees disconnections,
- transaction migration to model mobility, and
- transaction proxies to handle recovery.

Finally, we have briefly reported on the status of our system and discussed performance criteria for testing the proposed techniques.

References

[1] Rafael Alonso, Daniel Barbara, and Hector Garcia-Molina. Data Caching Issues in an Information Retrieval System. *ACM Transactions on Database Systems*, 15(3):359–384, September 1990.

[2] Rafael Alonso, Eben M. Haber, and Henry F. Korth. A Mobile Computer Interface for Heterogeneous Databases. In *Proceedings of the RIDE-IMS'93 Workshop*, Vienna, Austria, April 1993.

[3] Rafael Alonso and Henry F. Korth. Database System Issues in Nomadic Computing. In *Proceedings of the 1993 SIGMOD Conference*, Washington, D.C., May 1993.

[4] Baruch Awerbuch and David Peleg. Concurrent Online Tracking of Mobile Users. In *Proceedings of the ACM SIGCOMM Symposium on Communications, Architectures and Protocols*, pages 221–233, September 1991.

[5] Bharat Bhargava, Yinhe Jiang, and Jagannathan Srinivasan. O-Raid: Experiences and Experiments. In *International Conference on Intelligent and Cooperative Information Systems, Rotterdam, Holland*, May 1993.

[6] Bharat Bhargava and John Riedl. The Raid Distributed Database System. *IEEE Transaction on Software Engineering*, 15(6), June 1989.

[7] Panos K. Chrysanthis. Transaction Processing in Mobile Computing Environment. In *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems*, pages 77–83, Princeton, New Jersey, October 1993.

[8] Daniel Duchamp, Steven K. Feiner, Jr Gerald and Q. Maguire. Software Technology for Wireless Mobile Computing. *IEEE Network Magazine*, pages 12–18, November 1991.

[9] George H. Forman and John Zahorjan. The Challenges of Mobile Computing. *IEEE Computer*, 27(6), April 1994.

[10] Jim Gray and Andreas Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, 1993.

[11] Object Management Group. The Common Object Request Broker: Architecture and Specification. OMG Document Number 91.12.1, December 1991.

[12] D. Hayden. The New Age of Wireless. *Mobile Office*, May 1992.

[13] Yixiu Huang, Prasad Sistla, and Ouri Wolfson. Data Replication for Mobile Computers. In *Proceeding of 1994 ACM SIGMOD*, May 1994.

[14] Tomasz Imielinksi and B. R. Badrinath. Data Management for Mobile Computing. *SIGMOD Record*, 22(1):34–39, March 1993.

[15] Tomasz Imielinksi and B. R. Badrinath. Wireless Mobile Computing: Challenges in Data Management. Technical Report DCS-TR-296 and WINLAB TR-49, Rutgers, Dept. of Comp. Sciences, 1993. To appear in Communications of the ACM, 1994.

[16] T. Imielinski and B. R. Badrinath. Querying in Highly Mobile Distributed Environments. In *Proceedings of 18th VLDB*, pages 41–52, August 1992.

[17] John Ioannidis, Dan Duchamp, Jr Gerald, and Q. Maguire. IP-based Protocols for Mobile Internetworking. In *Proceedings of the ACM SIGCOMM Symposium on Communications, Architectures and Protocols*, pages 235–245, September 1991.

[18] James J. Kistler and M. Satyanarayanan. Disconnected Operation in the Coda File System. *ACM Transactions on Computer Systems*, 10(1):3–25, February 1992.

[19] John R. Nicol, C. Thomas Wilkes, and Frank A. Manola. Object Orientation in Heterogeneous Distributed Computing Systems. *IEEE Computer*, 26(6): 57–67, June 1993.

[20] Evaggelia Pitoura and Bharat Bhargava. Dealing with Mobility: Issues and Research Challenges. Technical Report TR-93-070, Purdue University, Dept. of Comp. Sciences, 1993. Available through the WWW, http://www.cs.purdue.edu/people/pitoura.

[21] Evaggelia Pitoura and Bharat Bhargava. Maintaining Consistency of Data in Mobile Distributed Environments. Technical Report TR-94-025, Purdue University, Dept. of Comp. Sciences, 1994.

[22] Amit Sheth and Marek Rusinkiewicz. Management of Interdependent Data: Specifying Dependency and Consistency Requirements. In *Proceedings of the Workshop on the Management of Replicated Data*, pages 133–136, Houston, Texas, November 1990.

[23] Carl D. Tait and Dan Duchamp. Service Interface and Replica Management Algorithm for Mobile File System Clients. In *Proceedings of the First International Conference on Parallel and Distributed Information Systems*, pages 190–197, 1991.

[24] C.J. Taylor. A Status Report on Open Distributed Processing. *First Class (Object Management Group Newsletter)*, 2(2):11–13, June/July 1992.

[25] Fumio Teraoka, Yasuhiko Yokote, and Mario Tokoro. A Network Architecture Providing Host Migration Transparency. In *Proceedings of the ACM SIGCOMM Symposium on Communications, Architectures and Protocols*, pages 209–220, September 1991.

[26] Yongguang Zhang and Bharat Bhargava. Wance: A Wide Area Network Communication Emulation System. In *Proceedings of IEEE Workshop on Advances in Parallel and Distributed Systems*, pages 40–45, Princeton, NJ, October 1993.