MOBILE COMPUTING and DATABASES: ANYTHING NEW?

Margaret H. Dunham
Dept of Computer Science and Engineering
Southern Methodist University
mhd@seas.smu.edu

Abdelsalam (Sumi) Helal
Dept of Computer Science
Purdue University
helal@cs.purdue.edu

1 Introduction

Recent advances in hardware technologies, such as portable computers and wireless communication networks, have led to the emergence of mobile computing systems. No one challenges the idea that mobile computing offers many opportunities for research within the Computer Science area. However, one could ask are there really any new database problems introduced when a mobile computing environment is assumed. We feel that the status of data management in mobile computing is similar to that of distributed data management versus centralized data management in the late 60s. Namely that many of the issues are the same, but the solutions are different. This analogy has been raised by others [1, 12]. We use this as the basis by which we answer the above question. We concentrate on discussing the differences between data management solutions in a mobile computing environment and those in a distributed database environment.

The purpose of this paper is twofold: to spawn further interest in mobile computing research and to convince the skeptics that there are new research topics in mobile computing worthy of further examination.

2 Mobile Computing Environment

Figure 1 shows the existing (and widely accepted) architectural model of a system that supports mobile computing [7, 8, 9, 4]. The model consists of stationary and mobile components. The only mobile component is the mobile unit. A *Mobile Unit* is a mobile computer which is capable of connecting to the fixed network via a wireless link. Stationed hosts are connected together via a fixed high-speed network (Mbps to Gbps). Components in this fixed network are classified as either fixed hosts or base stations. A *Fixed Host* is a computer in the fixed network which is not capable of connecting to a mobile unit. A *Base Station* is capable of connecting with a mobile unit and is equipped with a wireless interface. They are also known as *Mobile Support Stations*. Base stations, therefore, act as an interface between mobile computers and stationed computers. The wireless interface in the base stations typically uses wireless cellular networks. The wireless interface can also be a local area network, of which NCR WaveLan is an example. Current cellular technology offers a limited bandwidth in the order of 10 Kb/s (Mobidem offers 8 Kb/s), whereas current wireless LAN technology offers a bandwidth in the order of 10 Mb/s (WaveLan offers 2 Mb/s). While these numbers are most likely to change in the future, it is safe to assume that the network bandwidth will remain a major limitation and a performance bottleneck for nomadic system design in the near future.

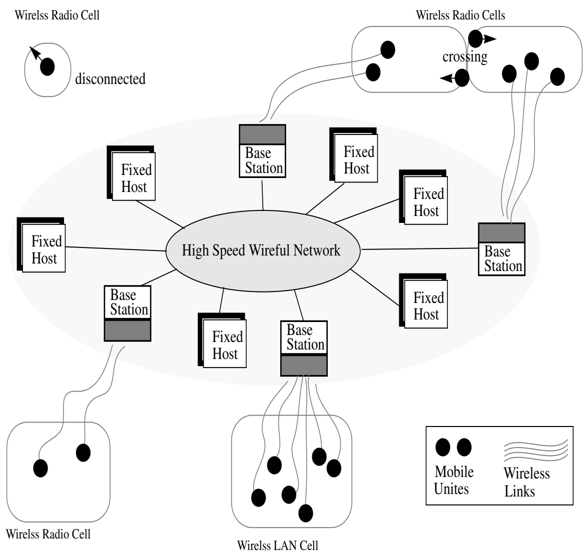

Figure 1: Architecture for Mobile Systems

There have been several previous surveys of mobile computing and what challenges it offers for database research [1, 5, 6, 7]. Imielinski and Badrinath have produced three excellent introductions into the topic [5, 6, 7]. They categorize research into the areas of mobility (locating users, queries stated on location dependent data, replication of data in the mobile units), disconnection (keeping the data cache at the mobile unit consistent, handoff and recovery issues related to transaction processing), new data access modes related to the use of the wireless medium and problems caused by battery life on the mobile unit, and scale. Alonso and Korth concentrate on the impact that mobile computing (or nomadic computing) has on various database processing activities [1]. The central issues discussed center around query optimization, transaction models, and security. They also include a discussion of applications of mobile computing.

3 Examples

We provide some examples of the types of database applications which will exist in mobile computing. With each we briefly discuss what differences these applications have from traditional distributed applications.

Example 1: Some of the most commonly cited examples of mobile computing applications are location dependent queries. Imagine a family driving down the interstate on a vacation. Future location dependent databases will maintain information about motels, restaurants, and other needed local services (doctors, hospitals, etc.). Location dependent queries access this information to answer such questions as: Where is the closest motel with a pool? How do I get to the closest hospital? Unlike queries in a centralized or distributed database environment, the response to these queries depends on the location. The same question may be asked repeatedly (E.g. Where is the closest public restroom?) with a different response everytime it is asked. Its response is thus location dependent.

The research issues involved in this example include: (1) Mobile (or location-sensitive) query languages, specifically SQL extensions, (2) Mobile database design, especially those issues related to the automatic maintenance of metadata that maps data sets to locations, and (3) efficient yellowpage archiving that will allow data (about services) and its associated location to be dynamically registerd and discarded.

Example 2: Currently salesmen frequently use laptops to maintain information about clients and orders. When a salesman arrives at a client's location he may request a display of that information (You wouldn't want to forget to mention your customer's favorite football team). Although similar queries could be addressed in a distributed database system, the query would be sent to the site in the fixed network where the information is maintained. In a mobile computing database environment this same query could at some times be directed to that fixed site while at other times directed to a cached local copy of the information. This illustrates several differences from the distributed version. Based on the state of the mobile unit (connected or disconnected) a query or transaction may be executed differently. Secondly, the caching of the data on the mobile unit, while like data replication in a distributed database environment, adds many new challenges for solutions.

Some of the research issues related to this example are: (1) How can we specify the degree of consistency between the mobile application cache and the stationary copy? Should this be implemented at the system-level or at the programming language level? The application-aware adaptation described in [10] leans more towards avoiding system-level solutions. (2) Migration of data into the mobile host will be necessary for extremely long-lived applications. Migration is a dynamic data redistribution, not a form of caching. How can data be migrated into the mobile application? This is not achievable by simply invalidating the primary copy (stationed copy). And even if migration is successful, would access to an already migrated data be allowed only locally? If not, we end up with a mobile application, mobile server model, whose behavior is very difficult to predict. (3) How can dirty caches be synched with the system under severe communication bandwidth restrictions? Perhaps *fax*-mode transmission, or message piggybacking would help in utilizing such poor bandwidth, whenever the mobile user communicates, or whenever the mobile unit is powered on. However, piggybacking may be unnecessarily slow. Adaptability will perhaps be a key in achieving cache synching efficiently.

Example 3: This is a futuristic example that is related to Example 2. We base the example on *Los Angeles* like traffic, where carpoolers sometimes spend over one hour commuting to or from work. On the road, an employee may use her labtop to connect to her corporate database server and start processing customer requests, or generating invoices. If this application is realized, a two hour increase in productivity will be achieved daily. We believe this will be a successful mobile application if performance and reliability are guaranteed. Otherwise, it may have an opposite effect. Caching and disconnection management therefore will be crucial to this application. The need for *group caching* optimization may arise in this application, where communication messages between the commuter bus and the company may be grouped together in multicast messages. Work in support of this optimization can be found in [2].

Example 4: This example is an extension of one found in [3]. In an automobile insurance agency, an insurance adjuster must physically examine each damaged automobile and provide a cost estimate for needed repairs. To provide this estimate she needs to access information about the automobile, any police reports concerning the car accident in which the vehicle was damaged, information about previous insurance claims from this individual, and the current value which the automobile has as found in the "Blue Book". We first examine how this Adjustment Transaction would be accomplished in the world of today (perhaps using a distributed system). We then examine how it could be performed in the mobile world of 2005. Figure 2 illustrates this example. In the world of 1995, the adjuster receives the request to perform the adjustment transaction either in written form from her supervisor or in electronic form when logging into her PC/terminal in her office/home. Prior to examining the car, the adjuster must gather information about the insuree, automobile, and report. Some of the information is gathered via phone calls, some must be requested in person at the appropriate location, and others by requesting it from internet sites. The adjuster then drives to the location of the automobile and physically examines it. As she does so, written notes are made concerning the damage. The adjuster then drives back to her office and executes the adjustment transaction. All of the information gathered throughout the day is input via this transaction and the appropriate report is sent to her supervisor. Much of the work was performed off line while the needed information was being gathered Now, how do we envision that this work could be performed in the year 2005 using mobile transactions? First of all, much of the gathering of the needed data will be performed online. After the adjuster has examined the claims to be filed for that day, she determines which to evaluate first. Then she immediately begins the adjustment transaction. Logically, this transaction performs the same functions as that of the 1995 model, but the gathering of data is done by the transaction itself. Subtransactions are generated to obtain all of the needed data automatically. As they are executed, the data needed for the final report is collected. At any time the adjuster may decide to view the data at her mobile computer screen. Note that the processing is occurring as she is enroute to the location of the damaged automobile. By the time she arrives at the car location, all of the needed data is there. When she examines the car she enters all the needed information into the mobile unit which automatically updates the data in the fixed network. When she leaves the scene, she has finished processing the claim. She can then begin on the next adjustment transaction as she drives to the location of the next damaged automobile. On the way to examine the second automobile, however, the adjuster decides to stop for lunch. To conserve power, she turns off her mobile unit. After lunch she turns the unit back on and resumes the active transaction.

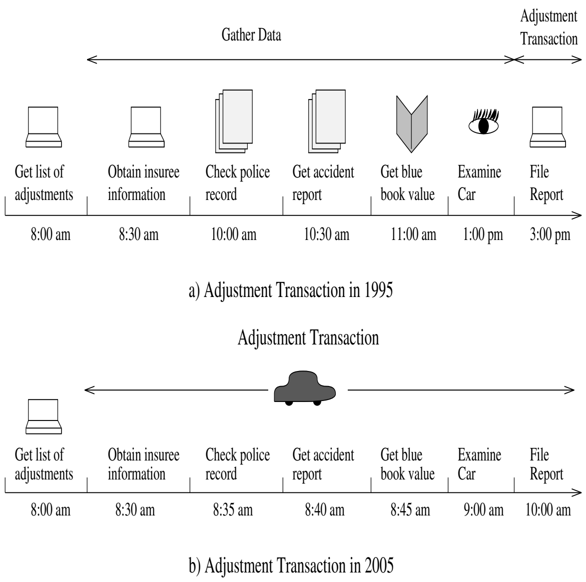

Figure 2: Mobile Transaction Example

The use of the mobile unit has had several impacts on the processing of the claim. First of all the claims processing has been speeded up and overall efficiency of the the adjuster has been increased. Absolutely no paperwork has been involved. Secondly, the length of time that the transaction is active has increased. In addition we see several new problems which are introduced. First, the transaction is extremely long lived. In addition, the user wishes to keep the transaction active even though she has disconnected from the system. The frequency of network partitioning in the mobile environment definitely increases. An interesting twist on this is that this voluntary disconnection should not be viewed as a failure. On the other hand some involuntary ones may need to be dealt with as failures. These issues all complicate transaction processing and recovery.

4 Anything New?

We view a mobile DBMS computing environment as an extension of a distributed system. Özsu and Valduriez have provided an excellent classification for distributed DBMSs based on the system characteristics of autonomy, distribution, and heterogeneity [11]. Figure 3 extends their classification to include mobile DBMS systems. To their classification we have added an extra point on the distribution axis. This is because a mobile computing system must include a fixed network (see Figure 1) which is a distributed system. A mobile computing system can thus be viewed as a dynamic type of distributed system where links between nodes in the network change dynamically. These intermittent links represent the connection between the mobile units and the base stations to which they are connected. We thus categorize a mobile computing database environment as: a Mobile Heterogeneous Multidatabase System.

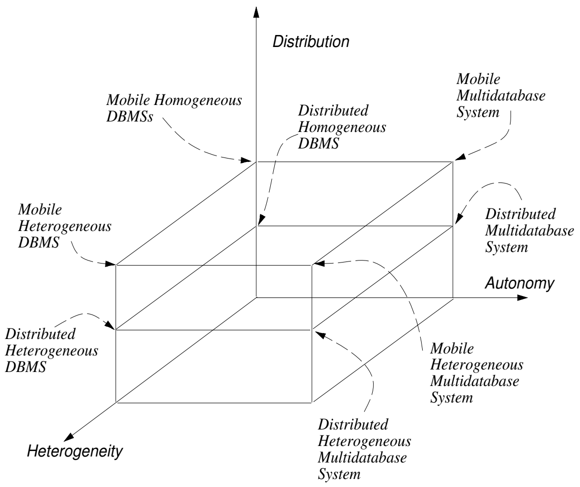

Figure 3: A Classification of Mobile Database Systems (adapted from Figure 4.4 in [11])

The relationship between distributed and mobile computing systems was discussed at the Workshop on Mobile Computing Systems and Applications in December 1994 [12]. They "explored the claim that mobile computing was merely a special case of distributed computing" [12, p12]. They felt that while many of the problems were the same there were some differences. The difference between the goal of "location transparency" in distributed applications and "location awareness" in mobile applications was a major point raised. We saw this in the first example above. Alonso has pointed out that another difference is the cost/performance aspects [1]. This forces many of the best solutions to distributed database problems to be unacceptable solutions in the mobile computing environment. Our classification above as well as the examples given in this paper indicate that distributed computing is actually a special case of mobile computing. As such, some solutions for distributed computing problems will work in the mobile computing arena while others may not.

Data Mgmt Issue	**Difference with Mobile Computing**
Applications	• May be location dependent • Need to adapt to changes in system context
Transactions	• New models to capture mobility
Recovery	• Frequent network partitioning • Volunary MU shutdown is not system failure • Mobility may cause more logging • Techcniques to recover from disconnection during handoff
Replication	• Differ. consistency constraints • New techniques for MU cache update due to freq disconnect
Query Processing	• Location dependent • Different cost factors • Query responses returned to different location • Adaptive techniques needed
Name Resolution	• New global name strategy due to mobility & disconnect

Table 1: Differences Between Distributed Data Management and Mobile Data Management

In Table 1 we show some of the differences between solutions to traditional data management problems in a distributed database and those found in a mobile computing network. In a distributed environment, a DBMS needs to be able to recover from site, media, transaction, and communication failure. The same types of failure occur in a mobile environment. However, the frequency of most of them increase. These problems complicate recovery (which is already complicated due to the mobility of transactions). Site failure at the MU may be frequent due to limited battery power. In addition, a MU may suffer from frequent voluntary shut downs which should not be treated as system failures. Transaction failures may increase due to the possibility of problems during the handoff when the MU moves between cells. A MU failure creates a partitioning of the network which in turn complicates updating and routing algorithms. Another major difference lies in the transaction model. Unlike a distributed transaction, a mobile transaction is not identified by a home and remote sites. It is identified by the collection of sites it hops through. A distributed transaction is executed concurrently on multiple processors and data sets. The execution of the distributed transaction is coordinated fully by the system (including concurrency control, replication management, and atomic commit). The mobile transaction, on the other hand, is executed sequentially through multiple base stations, and on possibly multiple data sets, depending on the movement of the mobile unit. (A sub-execution of the mobile transaction at a given base station can be considered a distributed transaction). The execution of the mobile transaction is therefore not fully coordinated by the system. The movement of the mobile unit, to a great extent, controls the execution. This fact must be reflected in newer transaction models.

5 Summary

We have highlighted some of the problems which mobility adds to databases in a distributed environment. By viewing distributed databases as a special case of mobile databases, we have seen that some solutions for database problems in the distributed environment do not work well in a mobile environment.

The interested reader can find more information by visiting our home page at URL:http://www.seas.smu.edu/~mhd/moblinks.html

6 Acknowledgements

We wish to thank Alex Delis for providing the much improved version of Figure 3 which is seen in the paper.

References

[1] Rafael Alonso and Henry F. Korth. Database system issues in nomadic computing. In *Proceedings of the ACM International Conference on Management of Data*, pages 388–392, May 1993.

[2] Kenjiro Cho and Kenneth P. Birman. A group communication approach for mobile computing. *IEEE Operating Systems Bulletin*, 7(1), 1995.

[3] Panos K. Chrysanthis. Transaction processing in mobile computing environment. In *IEEE Workshop on Advances in Parallel and Distributed Systems*, pages 77–82, October 1993.

[4] T. Imielinski, S. Vishwanathan, and B. R. Badrinath. Data on the air: Organization and access. Technical Report DCS–TR, Department of Computer Science, Rutgers University, New Brunswick, NJ 08903, 1993.

[5] Tomasz Imielinski and B. R. Badrinath. Data management for mobile computing. *ACM SIGMOD Record*, 22(1):34–49, March 1993.

[6] Tomasz Imielinski and B. R. Badrinath. Wireless mobile computing: Solutions and challenges in data management. Technical Report DCS–TR–296, Department of Computer Science, Rutgers University, New Brunswick, NJ 08903, 1993.

[7] Tomasz Imielinski and B. R. Badrinath. Mobile wireless computing. *Communications of the ACM*, 37(10):19–28, October 1994.

[8] John Ioannidis and Gerhald Q. Maguire Jr. The design and implementation of a mobile internetworking architecture. In *Proceedings of the 1993 Winter USENIX Conference*, pages 491–502, January 1993.

[9] David Johnson. Ubiquitous mobile host internetworking. In *Proceedings of the Fourth Workshop on Workstation Operating Systems*. IEEE, October 1993.

[10] B. Noble, M. Price, and M. Satyanarayanan. A programming interface for application-aware adaptation in mobile computing. In *Proceedings for the Second USENIX Symposium on Mobile and Location-Independent Computing*, April 1995.

[11] M. Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems*. Prentice-Hall, Inc., 1991.

[12] M Satyanarayanan. Digest of proceedings: Workshop on mobile computing systems and applications-december 1994. *IEEE Computer Society Bulletin of the Technical Committee on Operating Systems and Application Environments*, 7(1):5–12, 1995.