On the Construction of Mobile Database Management Systems

Athman Bouguettaya
School of Information Systems
Queensland University of Technology, Gardens Point Campus
Brisbane 4001, Queensland, Australia
athman@icis.qut.edu.au

Abstract

Managing the infoglut in this information age requires more than the traditional and fixed computing environment in order to keep up with fast changing technologie. Cellular and mobile communications offer flexibility in accessing and manipulating information without restricting users to specific locations. While this technology is revolutionary in its approach to managing data, many issues remain to be addressed. In this paper, we focus on the database aspects of mobile computing. We provide an overview of the different issues mobile DBMSs must address for them to provide an adequate environment for database mobile users. We propose a minimal mobile DBMS architecture and suggest further research to bring this area to a mature state.

1 Introduction

Advances in cellular communications and satellite services will enable mobile users to access information across the globe. As the technology matures, millions of people will become mobile users [Col94] [Kob93]. Access to various information services will be through some form of portable computer like a *Personal Digital Assistant* or PDA. These personal assistants are equipped with wireless connection to the equivalent of the *Internet* [QH86]. In this context, some information will be *broadcasted*. Other types of information will be accessed *on demand*. Mobile palmtops will be evolving to require not only access to simple types of data like phone numbers but also access to full fledged database systems. These databases are expected to be autonomous and heterogeneous in nature. Wireless database access already exists though not in its matured state. For example, New York City officials check license plates from databases via a wireless–data network and have late payers' cars towed away immediately. A similar example is the rental car company *Avis* use of mobile palmtops to update a central database about cars check–in and check–out information. With the advent of *ubiquitous computing* [Wei93], addressing the issues related to cellular computing will become even more important. Ubiquitous computing aims at pushing computer devices as we traditionally know them, *out of the way*. This will mean that many tools that we use on a daily basis will be replaced by cellular

devices. For instance, the little *post–it* tabs will be replaced by cellular replicas which will communicate with other different cellular devices.

Several problems hamper taking full advantage of the revolutionary wireless technology. Some of the problems are technological while others are technical. The technological barriers are here to stay for a long period of time [IB93b] [MDC93] [FZ94] [IB93a]. For instance, the problem of battery life expectancy is not expected to increase substantially in the foreseeable future [AG92]. The current average battery capacity is 2–3 hours. As the move towards miniaturization continues, the palmtop screens will also get smaller. This poses a fundamental problem in terms of representing textual as well as multimedia information on the palmtop screens [FZ94]. No breakthrough in terms of data visualization hardware is expected (like foldable screens) [AK93]. *Virtual reality* head gears have been investigated and have been found to have some serious drawbacks, including the inconvenient head gear, low resolution, and requirement of dim lighting [DKM91].

It is also argued that mobile computing will introduce dramatic changes in the way we know database systems. These changes can be compared to the effect that distribution and parallelism has had on database systems [IB93b]. This in fact means that solutions to many areas of database systems have to be revisited to retrofit the mobile environment. In essence, mobility introduces some fundamental problems to designing database systems. These are location management and scalability. These problems have a direct impact on the design and architecture of the different mobile database system components. The scale of the network has an important impact on how solutions to sharing information ought to be designed. We tried to address the scale issue and other related issues in non–mobiles networks of autonomous and heterogeneous databases [MBP95] [BKM95]. Some of these issues get compounded by the introduction of wireless databases.

The *Datacycle* project takes a novel approach to manipulate data [BGH+92]. In this approach, the whole database is broadcasted over a high speed network (52 MB/s) every 0.6 sec. Data *filtering* is achieved through dedicated *VLSI data filters* within multiple access managers. The major advantage of this approach is that *no* access method is required. While this research shares some commonalities with mobile databases, it does not optimize access based on *energy consumption.* This criterion makes some of the research results of limited benefit to mobile databases.

Mobility

It is important to note that mobility as such has an impact on both the mobile and fixed components of the network. Let us note that a fixed network allows communications only from fixed locations. However, a mixture of fixed and mobile components allows communications from *any* physical location. This limitless capacity to move freely and have access to the data of interest poses new challenges in terms of data management. Unlike fixed networks, the location of a mobile station (or user) becomes of prime importance since it is a varying component of the system. Therefore the system needs to manage this piece of data (user location) like any other database item, to efficiently process queries. Decisions that have to be made include data replication, organization, and access [IB93b]. Station mobility has a direct impact on network configuration. Cellular protocols will rely on this information to carry out efficient processing. Mobility complicates data manipulation by making network configuration a dynamic (even unpredictable) criterion. Because of frequent disconnections, the system has to react accordingly, and keep the system always consistent and functional [SlKM+93].

Scale

As mentioned earlier, the aforementioned problems get more complicated when the scale becomes large. It is expected that mobile palmtops (miniature database clients) will increase by millions in the next few years [IB93b]. This will engender challenging problems as to how databases ought to be managed in a highly mobile environment. For instance, frequent mobile disconnections (e.g. insufficient power) results in questions

being asked as to how a transaction manager copes with high numbers of transactions failures. Questions regarding crash recovery will also have to be addressed. These issues and related ones will be covered in the next sections.

2 Major Research Issues

We identify few research issues in mobile databases, namely: Location Management, Access Methods, and Query Processing. We omit other important issues like networking protocols, caching, and user interfaces for the sake of staying focused. [Joh93] and [IM93] propose two different schemes for mobile internetworking. A good overview on the caching issues can be found in [BI94]. [FZ94] and [AHK92] summarize the issues related to mobile computers. Below is an overview of the major research issues that we believe, lie ahead.

Architectural Assumptions

Mobile computing requires the cooperation of both *fixed* location and *mobile* hosts. The global architecture relies on this basic premise. The fixed location hosts may or may not have not a wireless interface. In case a fixed host has a wireless interface, it is called a *mobile support station* [IB93b]. The mobile host is called *mobile unit*. In this architecture, mobile units communicate either directly to mobile support stations or indirectly to fixed host through mobile support stations. The mobile support station stores applications software that can be downloaded by the mobile unit (e.g. PDA) and executed there.

The overall architecture will rely on the available underlying physical cellular architecture. In this latter case, a geographic area is partitioned into cells. A base station covers a cell. Its wireless interface enables it to communicate with mobile units. Its fixed location enables it to communicate with the fixed network. A cell typically spans 1 to 2 miles of a geographic area [Rou93]. Since each mobile unit will have a home station that knows about its location, the top level layer will consist of a hierarchy of *Location Servers* used to store information about the location of mobile users as well as other types of useful information. In fact, the location servers are mobile switching offices that are an integral part of the phone system. Therefore, the hierarchy of location servers (tree structure) mimics the typical architecture of the switching offices of the phone system.

Unlike fixed distributed systems, the configuration of wireless networks is highly dynamic in nature due to the mobility of users. Therefore, keeping track of where a mobile user is, is problematic. To address this problem, mobile users are uniquely assigned an *id* that refers to a *Home Station*. This latter either knows the exact cell the mobile user is currently in or a set of cells the mobile user is in. To ease the search, a mobile user may temporarily be assigned to a non–home station. In case of a call, a message is multicast (message *paging*) to a set of base stations under a location server. When a mobile user moves from one cell to another, a new frequency is assigned to the base station. The transition from one cell to another is called a *handoff*.

Location Management

Mobility of stations makes it necessary to keep track of its location to enable the correct processing of queries. Therefore, this data becomes one of the focuses of system performance. Indeed, establishing a contact with a station requires prior knowledge of its location. This potentially means that the location becomes a piece of data that can be both *queried* and *updated*. The search for this piece of data should be as efficient as any other queried data. Queries involving this data can be as complicated as asking for a specific policeman in a specific geographic area.

There are several ways of managing a location [IB93b] depending on what the assumptions are. For

instance, if a station A request the location of B, should we assume that A has no knowledge of B's location, and hence let it search the whole hierarchy of servers? Or alternatively, should we assume B leaves its address to some server? Three ways to query a station location have been identified. The first approach is based on a home location server. In this approach, the home location server always knows the location of the requested station. The second approach is based on some local servers knowledge of the requested station location. If not, a home server is queried. The third approach is based on mobility history. In this scheme, the requested station leaves its location in some places where it is likely to be queried. This way, if the requesting station is located in these areas, a local server gives the location of the requested station. If not, the requesting station could either contact a home server or do an expanding search starting from its geographical location.

The above scheme relies on the mobile user informing some location server(s) about its location. A study [MHAO92] shows that these messages (requests for user locations) start outstripping the actual messages exchanged when the number of mobile users is large. Therefore, the above scheme does not scale well. The idea is not to let a server know a mobile user changes cells every time, but rather to find a way of doing this as infrequently as possible while carrying the usual message exchange with the greatest accuracy possible. This will in fact mean that servers will not keep a precise location (incomplete information). The key issue is to devise the best alternative and this will be based on the mobility patterns of the mobile user. These pieces of information are either gathered while users move around and used later; or alternatively, users could give schedules (profile) of their future moves. This will enable the system to make some educated guesses about the current location of the mobile user. A relatively good scheme is to associate a partition of either location servers or base stations to a schedule in order to minimize the number of updates and messages multicast (paged) [IB92]. A more general approach locating mobile users is described in [AP91]. This approach is based on two sets associated with each node in the network. The first set is the *Write* set. The second set is the *Read* set. The *Write* set contains the set of vertices that should be notified whenever it becomes the host of a mobile user. The *Read* set of those notified nodes will get updated with the notifying nodes. In order to find a mobile user, a node *Read* node is intersected with the *Write* set of other nodes. The main advantage of this approach is its non–use of any *a–priori* knowledge about users mobility. However, further experiments are warranted to narrow down the domain of applicability of each approach.

Access Methods

Traditional databases need access methods to efficiently access and update data. In the case of cellular computing, portable machines need to efficiently access data both locally and "intercept" it from the waves. Typically, data (likes stock prices) will be broadcasted from fixed databases. The challenge for portable computers is to use as little energy as possible while *filtering* the information from waves. This supposes that the mobile unit will not be listening *continuously* so to save some energy. However, we would like the effect of this *sporadic* listening to be as if *continuous* filtering was taking place. Therefore, the criteria for data access are *access time* and *filtering time*. The latter criterion brings new issues that are intrinsic to the use of mobile units.

if no access method is broadcasted with the data, the mobile unit will spend an exorbitant amount of time filtering information and thus draining its energy in a short amount of time. Therefore, an access method will always be needed to efficiently filter data that is broadcasted. In [IVB94], two access methods are compared: indexing and hashing. In addition, they consider two types of data propagation: data broadcasting and *on–demand* requests. This practically means the data medium storage is on the *air*.

Considering both access and filtering times is a tricky issue as the optimization of one access may conflict with the other. Therefore, the issue is to find the best trade–off. For instance, minimizing filtering time will mean extending the access time to find the right information. On the other hand, minimizing the access time means extending the filtering time. The trade–offs will primarily rely on users preferences. For instance, users with high powered laptops would trade filtering time for better access time. In contrast, a small powered laptop would probably trade access time for a lower filtering time. An approach based on a distributed

indexing scheme is described in [IVB93]. This approach mainly focuses on optimizing filtering time without giving much importance to access time. All the approaches described so far overlook the important problem of broadcast *delays*. The basic assumption is that information stored in buckets gives hints about other incoming buckets. Depending on the delay and filtering time, these hints can become useless as the information of interest may have already "passed by". More powerful structure and algorithms are needed to address this issue.

Query Processing and Optimization

We identify two types of queries. The first type of queries deals with general kinds of queries involving the content of databases. The second type of queries involves the location of mobile users. In the former case, we are more interested in *Continuous Queries* [AB92] [TGNO92]. Because of the mobility, new issues arise for query processing.

Continuous queries are semantically equivalent to queries that are being *continuously* executed against the database. Many commercial databases will be broadcasting their content *on the air*. PDAs will pick up the information from that data medium. For this to happen, the mobile computer will need to scan waves *constantly*. The major drawback is the limited energy resources (battery) of the mobile computer. Therefore, the query processing should be *energy* efficient. In that respect, the cost functions used by the optimizer will include energy consumption. This is tightly dependent on the mobile configuration as well as on the resource utilization. [AG92] looks at different architectures (stand alone operations and different types of client/server operations) to design a new cost function to generate the most efficient plans.

As mentioned above, the location of mobile users can be the object of complex queries. The end result could either be the location in itself (object of interest) or a result to execute other dependent queries. An instance of the first case is the following query: "Find the number of mobile users in a cell." An instance of the second case is the following query: "Find the number of firefighting trucks in a specified cell." In the first query, the end result is the number of users. In the second query, the number of firefighting trucks in a specified cell will determine what other queries should be in order to face a life threatening situation.

The optimization problem is to find the best plan in terms of number of messages exchanged and the precision of the answer. If the information relied upon is extracted from location servers, there is a possibility that the answer is incorrect (case of partition based profiles). However, the communication cost is nil. In case of message multicast, the answer is exact. However, the communication cost may grow to be substantial. Figuring out the best execution plan requires using the best techniques in both spatial query processing and management of incompletely specified data. However, this alone will not be enough to address the problem of optimization. This problem gets complicated by the fact that the query optimizer has to *dynamically* take into account the cost of getting more precise data as it is running. Therefore, optimization uses a multidimensional set of criteria to determine the best strategy.

3 Mobile DBMS Construction Directions

The research outlined above is in its majority, still in its infancy. Experiments in this emerging technology are needed in order to provide a proof of concept for the proposed approaches. It is quite remarkable that in all the studies conducted so far, the latency aspect had been overlooked. In the case of access methods, the approaches rely on information being read from a bucket to determine whether other buckets contain the information of interest [IVB94]. However, this assumption may not be realistic, depending on the CPU speed, load, channel bandwidth, and other criteria. In this regard, the mobile computer may well *miss* the bucket in question and wait for a second broadcast. This introduces a level of *uncertainty* in all approaches described so far [IVB94].

Accessing Data

In general, the problem is to keep the number of bucket misses as low as possible while being as power efficient as possible. It should be noted that previous research has assumed a homogeneous cellular network of databases. In reality, databases will be *heterogeneous* and *autonomous*. It is more so in the world of cellular computing. In terms of access methods, different databases will use different policies. Mobile users should be able to deal with heterogeneous access methods. For instance, a mobile user should be able to access a database containing current stock prices in the New York stock market *and* access another database containing stock prices in the Tokyo stock market. These database will probably be heterogeneous in terms of data modeling and DBMS functionality. The key research question is to be able to seamlessly deal with heterogeneity of access methods. This practically means that databases will broadcast extra information along with the data. The extra information will act as *hints* from the broadcasting databases. In essence, mobile hosts need to know what type of access methods are being used to efficiently access data on the air. It is not yet clear how interleaving buckets that are accessed differently will impact on power efficiency. Experiments are needed to determine whether there is any impact in that regard.

Broadcasting vs. On–Demand

Research so far has focused on database broadcasting as a means of making data available to mobile users. It is expected that in the near future ubiquitous computing will make its way into offices where cells (or rather micro cells) will be of a few cubic meters [Wei93]. In this context, everyday tools will become mobile hosts that are able to query as well as to update data using cellular technology. This means that data will alternatively be available either by broadcasting or on–demand. Further studies are needed to determine optimal strategies for data access (broadcast viz. on–demand). Similar problems have been looked at in the Datacycle project [BGH+92] where data is filtered from a high bandwidth network. In this context, both *Read–Only* transactions as well as update transactions compete for access to data. An algorithm is described that will always guarantee schedules are serializable. This is the extent of the similarity. What makes the Datacycle project algorithms in this context inapplicable is the different underlying architecture. In the Datacycle project, the architecture is a *fixed* high bandwidth network. In the cellular architecture, the network is *mobile* and typically low bandwidth. Energy consumption is not an issue in the Datacycle project. It is a central issue in the mobile network architecture.

Mobile Host Transactions

Update transactions assume a two way communication between mobile hosts and a base station (database). This will mean that mobile hosts will be sending buckets to the fixed database. It is a well known fact that sending packets drains more power than receiving them [Rou93]. To optimize on energy, more powerful buffering as well as caching and logging mechanisms need to be developed. For instance, depending on data semantics and current power level, the buffer manager can either decide to send buckets to the fixed host or instead cache it for later shipping. [BI94] looks at different cache invalidation techniques in light of mobile hosts' disconnection performance. So far, no such combination has been proposed [AG92] [BI94]. It is worth investigating how caching could help energy efficient query optimization.

Crash Recovery

It is of utmost importance that database stay consistent in case of crashes or temporary unavailability

of hosts. It is expected that temporary *disconnections* will form the bulk of data unavailability cases. The disconnections are mostly mobile hosts dependent. They can happen either intentionally or unintentionally. For instance, the mobile host could have a triggering system that allows killing or stopping a process in order to save energy. This instance is considered as an intentional disconnection. Other instances of disconnections are unavailability due for example, to thick walls or laptop falls or out of range locations. The CODA project addresses these problems in a file system framework [SlKM+93]. It relies on server replication and hefty mobile hosts caches to implement the key element of their approach, *hoarding*. In contrast to databases, no broadcasting or multicasting is performed, thus making the approach of little significance to mobile databases.

Continuous queries are of particular interest in the case of cellular networks [AB92]. It is unreasonable to assume that mobile hosts will be running queries and filtering information on a continuous basis as this would drain power very fast. Instead, it seems that in this case, shipping the query to fixed database and using some form of caching will constitute a better alternative. The results are then shipped back to the mobile host. However this creates a bottleneck as well as problems of load balancing at fixed hosts. The solution seems to be a trade–off. Depending on some selected criteria, the processing may alternate between the fixed and mobile hosts. This approach is worth investigating to determine the right criteria and load balancing.

DBMS Architecture

A question that needs to be answered is what parts of a DBMS are required on a mobile host. As memory gets cheaper along with increased miniaturization, memory capacity will no longer be a criterion to decide what part of a DBMS should be resident in memory. Rather, there is a need to define what parts are *necessary* in a mobile environment. This problem is similar in some respect, to the different possible configurations in client/server databases [DMFV90] [DR92]. Not all standard (fixed) configurations make sense in the context of mobile databases. For instance, only a limited disk space will be available and most likely will be used as a swap area or a repository for logs. It practically means that total replication of a database is not reasonable. Another factor is the high power consumption of disk accesses. These accesses should be kept to a minimum along with other types of optimizations. We expect the necessary mobile DBMS to consist of a simplified transaction management system and a query optimizer. Instead of an traditional access and a data manager, a *filter* will take on the role of accessing data from the air.

References

[AB92] Rafael Alonso and Daniel Barbara. The characterization of continuous queries in general environments. Technical report, Matsushita Information Technology Laboratory, December 1992.

[AG92] Rafael Alonso and S. Ganguly. Energy efficient query optimization. Technical Report MITL-TR-33-92, Matsushita Information Technology Laboratory, Princeton, NJ, 1992.

[AHK92] R. Alonso, E. Haber, and H. F. Korth. A database interface for mobile computers. In *Proceedings of the 1992 Globecom Workshop on Networking of Personal Communications Applications*, December 1992.

[AK93] Rafael Alonso and Henry F. Korth. Database system issues in nomadic computing. In *ACM SIGMOD Conference*, pages 388–392, Washington DC, May 1993.

[AP91] Baruch Awerbuch and David Peleg. Concurrent online tracking of mobile users. In *Proceedings of the ACM SIGCOMM Symposium on Communication, Architectures, and Protocols*, October 1991.

[BGH+92] T. F. Bowen, G. Gopal, G. Herman, T. Hickey, K. C. Lee, W. H. Mansfield, J. Raitz, and A. Weinrib. The DATACYCLE architecture. *Communications of the ACM*, 35[12]:71–81, December 1992.

[BI94] Daniel Barbara and Tomasz Imielinski. Sleepers and workaholics: Caching strategies in mobile environments. In *ACM SIGMOD Conference*, Minneapolis, Minnesota, May 1994.

[BKM95] A. Bouguettaya, R. King, and S. Milliner. Resource location in large scale heterogeneous and autonomous databases. *Journal of Intelligent Information Systems*, 5(2), 1995.

[Col94] Narses J. Colmenares. The FCC on personal wireless. *IEEE Spectrum*, 31[5]:39–46, May 1994.

[DKM91] Daniel Duchamp, Steven K., and Gerald Q. Maguire. Software technology for wireless mobile computing. *IEEE Network Magazine*, pages 12–18, November 1991.

[DMFV90] Dave DeWitt, David Maier, P. Futtersack, and Fernando Velez. A study of three alternative workstation–server architectures for object-oriented database systems. In *Proceedings of the 16th VLDB Conference*, pages 107–121, 1990.

[DR92] A. Delis and N. Roussopoulos. Performance and scalability of client-server database architectures. In *Proceedings of the 19th International Conference on Very Large Databases*, Vancouver, BC, Canada, August 1992. Boxwood Press.

[FZ94] George H. Forman and John Zahorjan. The challenges of mobile computing. *IEEE Computer Magazine*, 27[4], April 1994.

[IB92] T. Imielinski and B. R. Badrinath. Querying locations in wireless environments. In *Wireless Communications: Futures Directions*. Kluweir Academic Publishers, October 1992.

[IB93a] Tomasz Imielinski and B. R. Badrinath. Data management for mobile computing. In *ACM SIGMOD Record*, volume 22[1], pages 34–39, March 1993.

[IB93b] Tomasz Imielinski and B. R. Badrinath. Mobile wireless computing: Solutions and challenges in data management. *Communications of the ACM*, January 1993.

[IM93] John Ioannidis and Gerald Q. Maguire. The design and implementation of a mobile internetworking architecture. In *USENIX Winter '93 Technical Conference*, January 1993.

[IVB93] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Data on the air - organization and access. In *Submitted to the 10th IEEE Data Enginering Conference*, June 1993.

[IVB94] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Power efficient filtering of data on the air. In *EDBT Conference*, 1994.

[Joh93] David B. Johnson. Mobile host internetworking using IP loose source routing. Technical report, School of Computer Science, CMU-CS-93-128, February, 1993.

[Kob93] Bennett Z. Kobb. Personal wireless. *IEEE Spectrum*, 30[6]:20–25, June 1993.

[MBP95] S. Milliner, A. Bouguettaya, and M. Papazoglou. A scalable architecture for autonomous heterogeneous database interactions. In *Proceeedings of the Very Large Data Bases Conference (VLDB)*, Zurich, Switzerland, September 1995.

[MDC93] Brian Marsh, Fred Douglis, and Ramon Caceres. Systems issues in mobile computing. Technical report, MITL-TR-50-93, Matsushita Information Technology Laboratory, Princeton, NJ, February 1993.

[MHAO92] Kathleen S. Meier-Hellstern, Eduardo Alonso, and Douglas Oniel. The use of SS7 and GSM to support high density personal communications. In *Third Winlab Workshop on Third Generation Wireless Information Networks*, pages 49–57, April 1992.

[QH86] J. S. Quarterman and J. C. Hoskins. Notable computer networks. *Communications of the ACM*, 29(10):932–971, October 1986.

[Rou93] *Special Issue on Wireless Communications.* Byte Magazine, McGraw Hill, February 1993.

[SlKM+93] M. Satyanarayanan, James l. Kistler, Lily B. Mummert, Maria R. Ehling, Puneet Kumar, and Qi Lu. Experience with disconnected operation in a mobile computing environment. In *Proceedings of the 1993 USENIX Symposium on Mobile and Location-Independent Computing*, Cambridge, MA, August 1993.

[TGNO92] D. B. Terry, D. Goldberg, D. A. Nichols, and B. M. Oki. Continuous queries over append-only databases. In *Proceedings of the ACM SIGMOD*, pages 321–330, June 1992.

[Wei93] Mark Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36[7]:74–84, July 1993.