Applications of Moving Objects Databases

Ouri Wolfson

Department of Computer Science, University of Illinois, Chicago, IL, 60607

Phone: (312)-9966770

Fax: (312)-9963422

wolfson@cs.uic.edu

Eduardo Mena

Department of Computer Science and System Engineering, University of Zaragoza,

50018 Zaragoza, Spain

Phone: +34 (976)-762340

Fax: +34 (976)-761914

emena@unizar.es

Applications of Moving Objects Databases

ABSTRACT

Miniaturization of computing devices, and advances in wireless communication and sensor technology are some of the forces that are propagating computing from the stationary desktop to the mobile outdoors. Some important classes of new applications that will be enabled by this revolutionary development include location-based services, tourist services, mobile electronic commerce, and digital battlefield. Some existing application classes that will benefit from the development include transportation and air traffic control, weather forecasting, emergency response, mobile resource management, and mobile workforce. Location management, i.e., the management of transient location information, is an enabling technology for all these applications. Location management is also a fundamental component of other technologies such as fly-through visualization, context awareness, augmented reality, cellular communication, and dynamic resource discovery. Moving Objects Databases store and manage the location, as well as other dynamic information about moving objects.

In this chapter we will present the applications of Moving Objects Databases (MOD's) and their functionality. The target readership are researchers and engineers working in databases and mobile computing.

Keywords: Mobile technologies, Wireless technologies, Spatiotemporal database, Data processing software

BACKGROUND

In 1996, the Federal Communications Commission (FCC) mandated that all wireless

carriers offer a 911 service with the ability to pinpoint the location of callers making emergency requests. This requirement is forcing wireless operators to roll out costly new infrastructure that provides location data about mobile devices. In part to facilitate the rollout of these services, in May 2000, the U.S. government stopped jamming the signals from global positioning system (GPS) satellites for use in civilian applications, dramatically improving the accuracy of GPS-based location data to 5-50 meters.

As prices of basic enabling equipment like smart cell phones, hand held devices, wireless modems, and GPS devices continue to drop rapidly, the number of wireless subscribers worldwide will soar. Spurred by the combination of expensive new location-based infrastructure and an enormous market of mobile users, companies will roll out new wireless applications to re-coop their technology investments and increase customer loyalty and switching costs. These applications are collectively called *location-based services.*

Emerging commercial location-based services include Mobile Resource Management (MRM) applications such as systems for mobile workforce management, automatic vehicle location, fleet management, logistics, transportation management and support (including air traffic control). These systems use location data combined with route schedules to track and manage service personnel or transportation systems. Call centers and dispatch operators can use these applications to notify customers of accurate arrival times, optimize personnel utilization, handle emergency requests, and adjust for external conditions like weather and traffic. Other examples of location based services are Location-aware Content Delivery that use location data to tailor the information delivered to the mobile user in order to increase relevance; for instance delivering accurate driving directions, instant coupons to customers nearing a store, or nearest resource information like local restaurants, hospitals, ATM machines, or gas stations.

In addition to commercial systems, management of moving objects in location based systems arises in the military, in the context of the digital battlefield. In a military application one would like to ask queries such as "retrieve the helicopters that are scheduled to enter region R within the next 10 minutes".

Moving Objects Databases (MOD's), which include the management of transient location information, is an enabling technology for all the above applications. MOD is also a fundamental component of other technologies such as fly-through visualization (the visualized terrain changes continuously with the location of the user), context awareness (the location of the user determines the content, format, or timing of information delivered), augmented reality (the location of both the viewer and the viewed object determines the type of information delivered to viewer), and cellular communication.

Location management has been studied extensively in the cellular architecture context. The problem is as follows. In order to complete the connection to a cellular user *u*, the network has to know the cell id of *u*. Thus the network maintains a database of location records (*key, cell-id*), and it needs to support two types of operations: (1) Point query, when a cellular user needs to be located in order to complete a call or send a message, e.g., find the current location (cell) of the moving object with key 707-476-2276, and (2) Point update, when a cellular user moves beyond the boundary of its current cell, e.g., update the current location (cell) of the moving object with key 707-476-2276. The question addressed in the literature is how to distribute, replicate, and cache the database of location records, such that the two type of operations are executed as efficiently as possible. Related questions are how frequently to update, and how to search the database. Many papers have addressed this question and two good surveys of the subject are Bhattacharya & Das (1999) and Pitoura & Samaras (2001).

However, the location management problem addressed by MOD's is much broader. The main limitations of the cellular work are that the only relevant operations are point queries and updates that pertain to the current time, and they are only concerned with cell-resolution locations. For the applications we discussed, queries are often set oriented, location of a finer resolution is necessary, queries may pertain to the future or the past, and triggers are often more important than queries. Some examples of queries and triggers supported by MOD's are: during the past year, how many times was bus #5 late by more than 10 minutes at some station (past query); show me the taxi cabs within 1 mile of my location (set oriented present query); retrieve the estimated location of truck #56 tomorrow at 8:00 am (future query); retrieve the trucks that will reach their destination within the next 20 minutes (set oriented future query); send me message when a helicopter is in a given geographic area (trigger).

In terms of location-based-services software development, the current approach is to build a separate, location-independent management component for each application. However, this results in significant complexity and duplication of efforts, in the same sense that data management functionality was duplicated before the development of Database Management Systems (DBMS's). To continue the analogy, we need to develop location management technology that addresses the common requirements, and serves as a development platform in the same sense that DBMS technology extracted concurrency control, recovery, query language and query processing, and serves as a platform for inventory and personnel application development.

In this chapter we describe the approach in building a general purpose location management system, i.e., a Moving Objects Database (MOD). Such a database serves as the repository that stores and manages location as well as other dynamic information about moving objects. The main topics that will be discussed are: location technologies

and applications, location modeling/management, and MOD architecture and functionality.

MOD APPLICATIONS

In this section we discuss the kind of applications that can be built on top of MOD technology.

Geographic resource discovery: A mobile user provides its (partial) future trajectory to a service provider, and expects the answer to queries/triggers such as: notify me when I am two miles away from a motel (in my travel direction) which has rooms available for under $100 per night. The service provider uses a MOD to store the location information of its customers, and answers their queries/triggers.

Digital battlefield: The dynamic location and status of the moving objects (tanks, helicopters, soldiers) in the battlefield is stored in a MOD that must answer queries and process triggers of various degrees of complexity (e.g., How many friendly tanks are in region X?).

Transportation (taxi, courier, emergency response, municipal transportation, traffic control, supply chain management, logistics). In these applications the MOD stores the trajectories of the moving objects and answers queries such as: which taxi cab is expected to be closest to 320 State street half an hour from now (when presumably service is requested at that address); When will the bus arrive at the State and Oak station? How many times during last month was bus #25 late at some station by more than 10 minutes?

Location (or Mobile) e-commerce and marketing: In these applications, coupons and other location-sensitive marketing information is fed to a mobile device (that presumably screens it based on the user profile, and displays it selectively).

Mobile workforce management: Utilities and residential/commercial service providers track their service engineers and the MOD answers queries such as: which service crew is closest to the emergency at 232 Hill street?

Context-awareness, augmented-reality, fly-through visualization: In these applications the service provider feeds, in real time, the relevant information to the current location of a moving user. For example, a geologist driving through a terrain can use its hand-held device to view the area she sees with the naked eye, but with additional information superimposed. The additional information is fed by the server and may include seismographic charts, images of the terrain taken at another season, notes made by other geologists about each landmark in the viewable terrain.

Air traffic control: Currently commercial flights take "highways in the sky", but when free flight (FAA, 2004) is instituted, a typical trigger to the air-traffic control MOD may be: retrieve the pair of aircraft that are on a collision course, i.e., are expected to be less than a mile apart at some time-point.

Dynamic allocation of bandwidth in cellular network: Cellular service providers may track their customers and may dynamically change the bandwidth allocation to various cells to satisfy changing customer density.

Querying in mobile environments: A Mobile Ad-hoc Network (MANET) is a system of mobile computers equipped with wireless broadcast transmitters and receivers which are used for communicating within the system. Such networks provide an attractive and inexpensive alternative to the cellular infrastructures when this infrastructure is unavailable (e.g., in remote and disaster areas), inefficient, or too expensive to use (Haas, 1998). Knowing the location of the destination-computer enables better and more reliable routing of messages. Thus, maintaining the trajectories of mobile computers in a MOD is an attractive alternative. However, in this case, the MOD is

distributed among the moving objects themselves, since a centralized solution defeats the MANET purpose.

Currently, commercial MOD products provide a very limited set of capabilities, and they focus on transportation, particularly fleet management systems. Companies marketing such systems include Mobitrac (MOBITRAC Inc, 2002), Qualcomm (Qualcomm Inc., 2002), and @Road (At Road Inc., 2004).

LOCATION TECHNOLOGIES

Location sensing methods fall into three categories, triangulation, proximity, and scene analysis. In triangulation, several signals originating from known sources are correlated by a processor, to determine the location of this processor. Global Positioning System (GPS) receivers are the best known implementation of this method (Leick, 2003). Such a receiver is a special purpose computer chip which costs less than $100, and is as small as one cm^2. It receives and triangulates signals from 24 satellites at 20,000 KM. It computes latitude and longitude with tennis-court-size precision. A Differential GPS is assisted by ground stations and achieves 2-3 feet precision.

The second location sensing method is proximity, where the location of a moving object is determined to be within a small distance from a sensor. For example, RFID (AIM, 2004) tags transmit a digital response when contacted by radio signals from close-by scanning devices. Then the location of the tag is known to be very close to that of the scanning device.

The last method is scene analysis, where the location of an object with known dimensions can be determined by analyzing its image produced by a camera with a known location.

MODELING BASED ON POINT LOCATION MANAGEMENT

A fundamental capability of location management is modeling of transient location information, particularly the location of mobile devices such as cell phones, personal digital assistants, laptops, etc. These devices are carried by people, or mounted on moving objects such as vehicles, aircraft, or vessels. The location information is updated by positioning technologies. In this section we describe a point location modeling technique; an alternative location modeling technique based on trajectory management can be found in the next section.

A straightforward approach that is used by existing industrial applications such as fleet management and Automatic Vehicle Location (AVL), is to model the location as follows. For each moving object, a location-time point of the form (*l, t*) is generated periodically, indicating that the object is at location *l* at time *t*. *l* may be a coordinate pair (*x*,*y*), or a cell-id. The point is stored in a database managed by a Database Management System (DBMS), and SQL is used to retrieve the location information.

This method is called point-location management, and it has several critical drawbacks. First, the method does not enable interpolation or extrapolation. For example, assume that a user needs to know which police officers were within one mile from the location of an emergency that occurred at 3 pm. This information can only be retrieved for the moving objects that happened to generate a location update at 3 pm If an object did not generate an update at 3 pm, then its whereabouts at that time are unknown. The problem is even more severe for extrapolation, i.e., if a future location is requested; for example, which field service employees will be closest to a customer location at 5 pm? This query cannot be answered by the point-location method, even though the future location of the service personnel can be estimated by being based on current work schedules.

The second problem of the point-location method is that it leads to a critical precision/resource trade-off. An accurate picture of the precise location of moving objects would require frequent location updates that consume precious resources such as bandwidth and processing power.

Finally, a third problem of this method is that it leads to cumbersome and inefficient software development. Specifically, location based services will require the development of a vast array of new software applications. Doing so on top of existing DBMS technology has several drawbacks. First, existing DBMS's are not well equipped to handle continuously changing data, such as the location of moving objects. The reason for this is that, in databases area, data are assumed to be constant unless they are explicitly modified. For example, if the salary field is $30K, then this salary is assumed to hold (i.e., $30K is returned in response to queries) until explicitly updated. This constant-until-modified assumption does not hold for the location of moving objects which changes continuously. The second drawback is that location based services applications need to manage space and time information, whereas SQL is not designed and optimized for this type of queries and triggers. For example, the query “retrieve the vehicles that are inside region R always between 4 pm and 5 pm” would be very difficult to express in SQL. Finally, the location of a moving object is inherently imprecise because the database location of the object (i.e., the object-location stored in the database) cannot always be identical to the actual location of the object. This inherent uncertainty has various implications for database modeling, querying, and indexing. For example, there can be two different kinds of answers to queries, i.e., the set of objects that “may” satisfy the query, and the set that “must” satisfy the query. SQL semantics cannot account for this difference.

An interesting observation is that the point location management is used for two

different cases. One in which the route of the moving object is known a priori (e.g., trucking fleet management, municipal transit), and the other in which such information is not available. For example, in location-aware advertising consumers usually cannot be assumed to provide their destination, and this is also the case for the enemy in digital battlefield applications. In other words, the information available a priori is not utilized for tracking, and it is not updated as a result of tracking.

MODELING BASED ON TRAJECTORY LOCATION MANAGEMENT

In this section we outline DOMINO's model (Wolfson, 1999) of a trajectory, explain how to construct it, and explain how it solves the problems associated with point location management. Let us observe that there exist alternatives to the approach here (see, for example, Guting *et al* (2000) and Sistla *et al* (1997)). If possible, we make use of a priori or inferred information about the destination of an object. For example, the destination can be inferred based on a motion pattern (e.g., the person travels to the office between 8 am and 9 am), or by accessing auxiliary information (e.g., a calendar may indicate a meeting at a given time and address).

The method proposed is called trajectory location management. In this method we first obtain or estimate the source and destination of the moving object. For example, the object starts in New York City at the intersection of 57th street and 8th Ave. at 7 am and heads for Chicago at the intersection of Oak and State streets. Then, by using an electronic map geocoded with distance and travel-time information for every road section, a trajectory is constructed.

Electronic Maps

Before defining the trajectory, let us define the format of an electronic map. An

electronic map is a relation where each tuple in the relation represents a city block, i.e., the road section in between two intersections, with the following attributes:

- *Polyline*: the block polyline given by a sequence of 2D *x,y* coordinates: $(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)$. Usually the block is a straight line segment, i.e., given by two (x,y) coordinates.
- *Fid*: The block id number

The following attributes are used for geocoding, i.e., translating between an (x,y) coordinate and an address such as "1030 North State St." (let us assume that the even range of numbers on the block is 1000-1100, and the odd range is 997-1103):

- *L_f_add*: Left-side-from street number (in the example,1000)
- *L_t_add*: Left-side-to street number (in the example, 1100)
- *R_f_add*: Right-side-from street number (in the example, 997)
- *R_t_add*: Right-side-to street number (in the example, 1103)
- *Name*: street name
- *Type*: ST or AVE
- *Zipl*: Left side Zip code
- *Zipr*: Right side Zip code
- *Speed*: speed limit on this city block
- *One way*: a Boolean One way flag.

The following attributes are used for computing travel-time and travel-distance.

- *Meters*: length of the block in meters
- *Drive Time*: typical drive time from the one end of the block to the other, in minutes

Such maps are provided by, among others, Geographic Data Technology Co. (GDT, 2004). An intersection of two streets is the endpoint of the four block-polylines. Thus

each map is an undirected graph, with the tuples representing edges of the graph.

Dealing with Trajectories

The route of a moving object O is specified by giving the starting address or *(x,y)* coordinate (start_point), the starting time, and the ending address or *(x,y)* coordinate (end_point). An external routine available in most existing Geographic Information Systems, and which we assume is given a priori, computes the shortest cost (distance or travel-time) path in the map graph. This path denoted P(O) is given as a sequence of blocks (edges), i.e., tuples of the map. Since P(O) is a path in the map graph, the endpoint of one block polyline is the beginning point of the next block polyline. Thus the whole route represented by P(O) is a polyline denoted L(O). For the purpose of processing spatiotemporal range queries, the only relevant attributes of the tuples in P(O) are Polyline and Drive-Time.

Given that the trip has a starting time, for each straight line segment on L(O), we can compute the time at which the object O will arrive to the point at the beginning of the segment (using the Drive-Time attribute). This is the certain-trajectory, or c-trajectory. Intuitively, the c-trajectory gives the route of a moving object, along with the time at which the object will be at each point on the route. More formally, a c-trajectory is a sequence of straight-line segments $(x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n)$ in 3-dimensional space. The c-trajectory means that when the object starts at a location having coordinates (x_1, y_1) at time t_1, it will move on a straight line at constant speed and will reach location (x_2, y_2) at time t_2, and then it will move on a straight line at constant speed and will reach location (x_3, y_3) at time t_3, etc. The c-trajectory is an approximation of the expected motion of the object in space and time. The reason it is only an approximation is that the object does not move in straight lines at constant speed.

However, given enough straight lines, the approximation can be accurate up to an arbitrary precision. The number of line segments on the trajectory has an important implication on the performance and precision of queries and triggers. Specifically, the performance increases and the precision decreases as the number of line segments decreases. We adjust and fine-tune the number of line segments on each trajectory by using a method that has been studied in computer graphics, namely line simplification (Agarwal & Varadarajan, 2000; Douglas & Peucker, 1973).

The c-trajectory is stored in the server database and in a computer carried by the moving object. At any point in time t between t_i and t_{i+1} the server can compute the expected location of the moving object at time t. Observe that this technique solves the first problem associated with point location management. Namely, trajectory location management enables both, location interpolation and extrapolation. The server can compute the expected location of the moving object at any point in time between the start and end times of the trip. For example, if it is known that the object is at location (x_5, y_5) at 5 pm and at location (x_6, y_6) at 6 pm, and it moves in a straight line at constant speed between the two locations, then the location at 5:16 pm can be computed at anytime, i.e., before 5:16 (extrapolation) or after (interpolation).

Finally, the trajectory (or the uncertain trajectory) is obtained by associating an uncertainty threshold u_i with the i^{th} line segment on the c-trajectory. The line segment together with the uncertainty threshold constitute an "agreement" between the moving object and the server. The agreement specifies the following: The moving object will update the server if and only if it deviates from its expected location according to the trajectory by u_i or more. How does the moving object compute the deviation at any point in time? Its computer receives a GPS update every two seconds, so it knows its actual location at any point in time. It has the trajectory, so by interpolation it can

compute its expected location at any point in time. The deviation is simply the distance between the actual and the expected location. More formally, a trajectory is a polyline $(x_1, y_1, t_1, u_1), (x_2, y_2, t_2, u_2), ..., (x_n, y_n, t_n, u_n)$ in 4-dimensional space.

At the server, the trajectory is maintained by revising it according to location-updates from the moving object, and according to real-time traffic conditions obtained from traffic web sites. We have developed a traffic incident model, and a method of identifying the trajectories affected by a traffic incident. Observe that determining whether or not a trajectory is affected by a traffic incident is not a simple matter, and it requires prediction capabilities. For example, suppose that according to the current information Joe's van is scheduled to pass through highway section *x* twenty minutes from now, and suppose that a web site currently reports a traffic jam on highway section *x*. Will Joe's expected arrival time at his destination be affected by this? Clearly it depends on whether or not the jam will clear by the time Joe arrives at highway section *x*. One can use historical information and various traffic models to make this prediction.

Observe that the agreement (namely the trajectory plus the uncertainty threshold) between the moving object and the server solves the second problem of point location management. Namely, the tradeoff between resource/bandwidth consumption and precision has been broken. In trajectory location management the location of a moving object can be computed with a high degree of precision, using a small number of location updates, or no updates at all. In particular, if the moving object is “on schedule”, i.e., it does not deviate from its prescribed trajectory by more than the uncertainty threshold, then no resources are consumed for updates.

Finally, let us observe that a trajectory can be constructed by being based on past motion in which an object used the point location management. Namely, the trajectory can be constructed from a set of 3D points $(x_1, y_1, t_1), (x_2, y_2, t_2), ... ,(x_n, y_n, t_n)$ that were

transmitted by a moving object using the point location management method. One can simply connect the points along the shortest path on the map, and then associate an uncertainty u_i with line segment i. The uncertainty u_i can be bounded given the maximum speed of the object and the known times of the two GPS points immediately preceding and succeeding the i^{th} line (Pfoser & Jensen, 2000). Or, the uncertainty u_i can represent the maximum error of the GPS receiver.

MOD ARCHITECTURE AND FUNCTIONALITY

A Moving Objects Database (MOD) consists of static geo-spatial and temporal information, some of which is updated in real time (see figure 1). The static information includes maps, profile information about moving objects, and motion plans (e.g. vehicle *v* starts at address *a*, and will make deliveries at addresses *b*, *c*, and *d*, in this order).

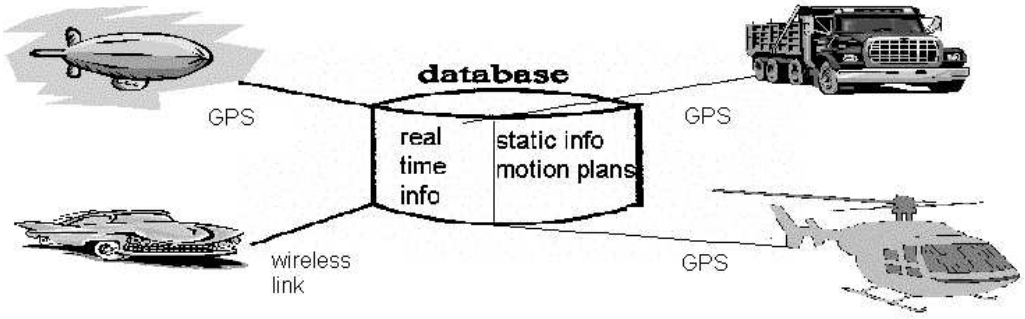

Figure 1: Moving Objects Database technology

The real time updates include current location and other information fed in by sensors. This is a conceptual model. In practice, location data may be incomplete, i.e., only partial trajectories may be available. Also, the database may be distributed rather than centralized. Moreover, the distribution may be among the moving objects themselves, with the extreme case being that each moving object stores its own location data.

Another generalization includes sensor data associated with the (time, location)

information. For example, fuel level, images of the environment, captured information from sensors in the instrumented infrastructure (e.g., availability of a parking slot in the neighborhood broadcast to the immediate vicinity by the slot), an accident in the vicinity indicated by a deployed air-bag sensor, etc.

MOD Architecture

A Moving Objects Database stores and manages the location, as well as other dynamic information about moving objects. It can be described as a three layer architecture, from the top to the bottom:

1. A software envelope that manages the dynamic aspects of the system.
2. A Geographic Information System (GIS) that provides different functionalities to deal with geo-spatial information.
3. A Database Management System (DBMS) which actually stores and manages the data

Thus, we wrap a DBMS and a GIS to build a platform for location-based-services application development.

Location data and other dynamic attribute values flow from moving objects to where the MOD resides (in general, in proxies or other elements of the wireless network). In the previous discussion we assumed a centralized architecture.

However, it is not realistic to think of a centralized proxy where the interesting data is sent from *all* the moving objects in the scenario (which could be crowded with moving objects!). In the same way as wireless networks are composed of a network of elements (proxies) that provide moving objects under its area with connectivity, we could think about a distributed approach to deal with location queries that involve moving objects highly distributed geographically. Thus, queries should be answered in an incremental

way (area by area) and the different answers joined recursively in order to obtain a final answer that would be presented to the user that issued the query. Agent technology (Milojisic et al, 1998) can be a good choice to implement an approach like the one described here: mobile agents will carry data (subqueries and their answers) to the right place, deal with disconnections and network delays, and, therefore, fit the dynamic nature of wireless networks quite well. Some work using this technology can be found in Ilarri *et al* (2002). The described distributed approach scales well, which is a very important feature in dynamic environments like wireless and *ad hoc* networks.

MOD Functionality

Here we demonstrate the query language and user interface for the DOMINO system (Wolfson, 1999). Figure 2 illustrates the answer to the query "Where are the vehicles at 12:35?"; the screenshot shows three trajectories on a map of Chicago, each with the 12:35 location indicated by a circle. The circle is centered at the expected location of the

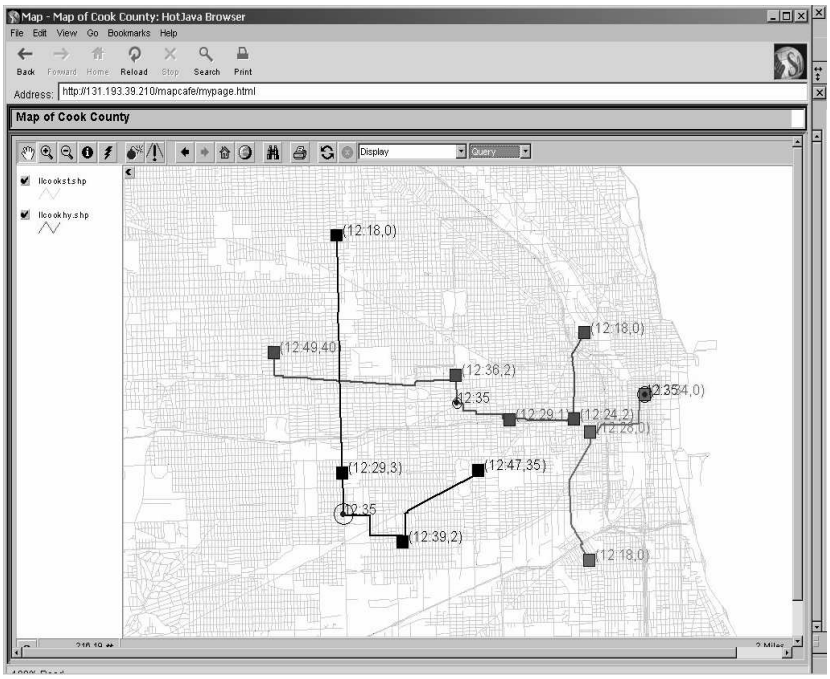

Figure 2: Query result showing all vehicles' locations at 12:35

vehicle, with the radius of the circle being equal to the uncertainty threshold. The squares of each trajectory indicate the stops of the trip. The label of each square indicates the time at which the moving object is (expected to be) at that location, at the time the moving object is stationary at that location. For example, the leftmost north-south trajectory starts at 12:18, arrives at the second stop at 12:29 and stays there for 3 minutes, arrives at the next stop at 12:39, and stays there for two minutes.

Figure 3 illustrates the query "Which moving object is closest to the star between 12:35 and 12:50?". The trajectories are displayed by the system, and the user marks the star on the map, and enters the time interval in the pull-down menu.

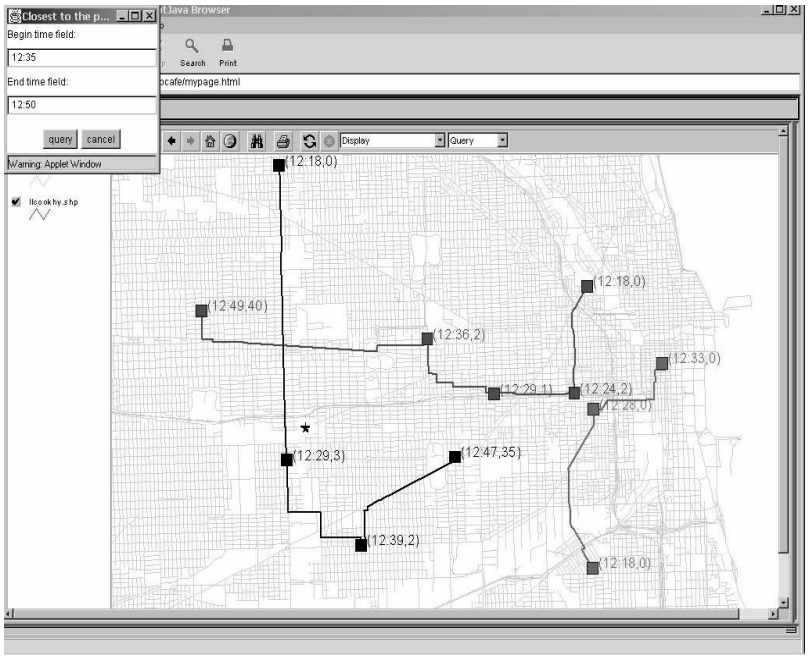

Figure 3: Which vehicle will be closest to the "star" between 12:35 and 12:50?

The answer of the query is illustrated in figure 4. Notice that, although black route (left north-south route) runs closer to the given point (the star), the answer to the query issued in figure 3 is a vehicle (indicated in the figure 4 as a circle) on the red route (east-west route). The reason is that, although vehicles following the black trajectory run closest to the star, they do not do so during the specified time interval.

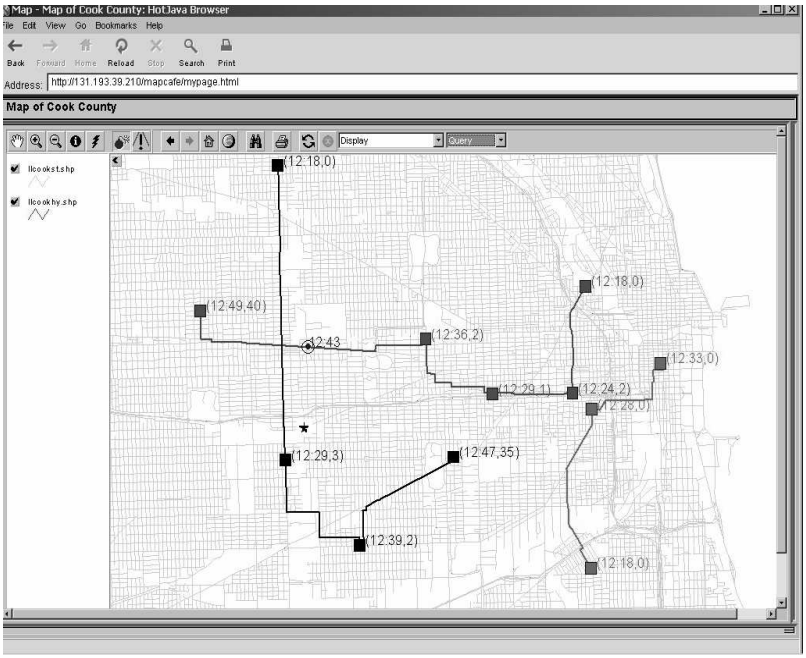

Figure 4: Answer: The vehicle (see the circle) on the red route

In Figure 5 we show another sample location query, this time is a set oriented query that requests the time at which the vehicles in the scenario enter a given area. The screenshot illustrate the query in which the user draws the area (shaded polygon) on the screen, and the system answers by showing the circles on each trajectory. Each circle touches the polygon, and is labeled with the time at which the vehicle is at the expected location.

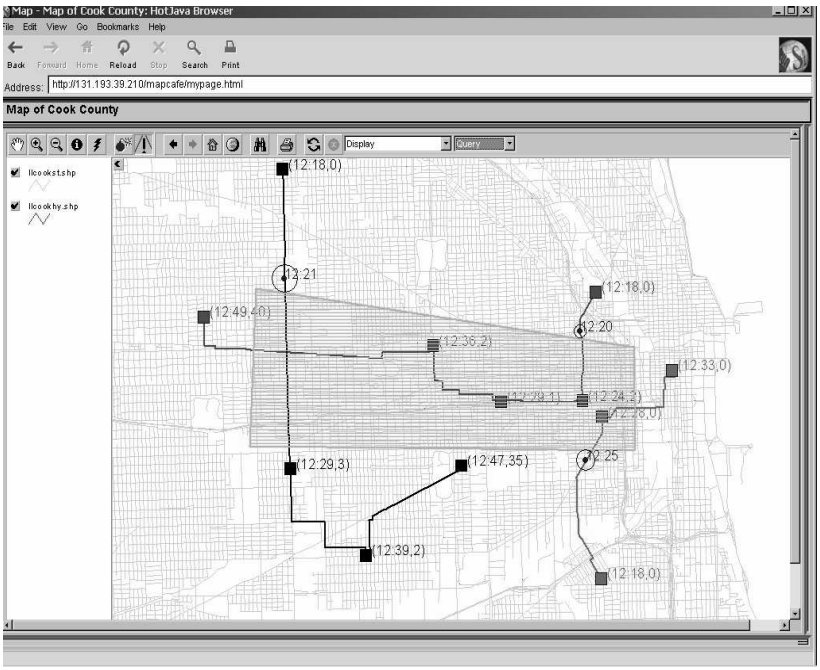

Figure 5: When will each vehicle enter the specified sector? (shown as shaded area)

CONCLUSIONS

We believe that the pervasive, wireless, mobile computing is a revolutionary development, and location based services and mobile resource management are some of the initial manifestations of this revolution. In this chapter we focused on location management, which is an enabling technology for a variety of applications and technologies related to this revolution. We discussed the research issues that need to be addressed in order to make location management a plug-in component in these applications and technologies. The first of these issues are location modeling. We discussed the drawbacks of existing approaches, and we proposed the trajectory as a four dimensional piece-wise linear function that captures the essential aspects of the moving object location. These aspects are two-dimensional space, time, and uncertainty. We also discussed moving objects database architecture and functionality, pointing out the need of considering a distributed approach for the location query processing. Then, we demonstrated the query language and user interface for the DOMINO system.

FUTURE TRENDS

Concerning future work, we believe that moving objects databases will become increasingly important, and we believe that DBMS's should be made the platform for developing moving objects applications. For this purpose, much remains to be done in terms of spatio-temporal query languages, support for rapidly changing real-time data, indexing, and distributed/mobile query and trigger processing with incomplete/imprecise location information. More specific open issues are enumerated in the following:

- A comparison of the existing indexing methods (for moving objects and queries) should be made in order to choose the most appropriate for each situation. The use of these index structures for join queries should be studied as well.
- Another issue is to extent the present work to handle uncertainty for moving objects that do not report their location; instead, their location could be sensed by possibly unreliable means. This is the case, for example, for enemy forces in a battlefield.
- Data mining techniques can be used to extract interesting information (for instance, trajectories of moving objects) from location data.
- Extensible and visual languages should be defined because, in this context, it is very important the way in which the users query the system and how the corresponding answers are presented; textual data are not enough as both queries and answers refer to data that should be displayed on a map.
- Concerning privacy/security, new methods are needed to assure that location data are only accessed by granted applications.
- Distributed approaches must be developed to process location queries in order to fit the natural distribution of mobile scenarios. Scalability becomes an important issue with the growing number of moving objects and queries.

REFERENCES

Agarwal P.K., & Varadarajan K.R. (2000). Efficient Algorithms for Approximating Polygonal Chains. *Discrete Comput. Geom.*, 23:273-291.

AIM: Association for Automatic Identification and Mobility (2004). Radio Frequency Identification (RFID) Homepage, http://www.rfid.org. 2004

At Road Inc. (2004). Http://www.@road.com. 2004

Bhattacharya, A., & Das, S. K. (1999). Lezi-Update: An Information-Theoretic Approcah to Track Mobile Users in PCS Networks, *Proceedings of the fifth ACM/IEEE International Conference on MobileComputing and Networking (MOBICOM99*), Seattle, WA, August 1999.

Douglas, D.H., & Peucker, T.K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Canad, Cartog*. 10(2):112-122, Dec. 1973.

FAA: The Federal Aviation Administration (2004). Http://www.faa.gov/freeflight. 2004

GDT: Geographic Data Technology Co. (2004). Http://www.geographic.com. 2004

Guting, R.H. & Bohlen, M.H. & Erwig, M. & Jensen, C.S. & Lorentzos, N.A. & Schneider, M. & Vazirgiannis, M. (2000). A Foundation for Representing and Querying Moving Objects, in *ACM Transactions on Database Systems Journal* (2000), 25(1), 1-42.

Haas, Z. J. (1998). Panel Report on Ad Hoc Networks - MILCOM'97. Mobile Computing and Communications Review, Vol. 2, No. 1, January 1998.

Ilarri, S. & Mena, E. & Illarramendi, A (2002). Monitoring Continuous Location Queries Using Mobile Agents. In Proceedings of Sixth East-European Conference on Advances in Databases and Information Systems (ADBIS'02), Bratislava (Slovakia). Springer-Verlag LNCS, September 2002.

Leick, A. (2003). GPS Satellite Surveying. John Wiley & Sons; 3rd edition (December 2003), ISBN 0471059307.

Milojisic et al (1998). MASIF, the OMG Mobile Agent System Interoperability Facility. In Proceedings of Mobile Agents'98, September 1998.

MOBITRAC Inc (2002). Http://www.mobitrac.com. 2002

Pitoura, E., & Samaras, G. (2001). Locating Objects in Mobile Computing. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No. 4, July/August 2001

Pfoser, D. & Jensen, C.S. (2000). Capturing the uncertainty of moving objects representations. *Proc. of the 12 Intl. Conf. on Scientific and Statistical Database Management*, 2000. IEEE Computer Society.

Qualcomm Inc. (2002). Http://www.qualcomm.com. 2002

Sistla, A. P. & Wolfson, O. & Chamberlain, S. & Dao, S. (1997). Modeling and Querying Moving objects, In *Proc. of the International Conference on Data Engineering* (1997) pp. 422-432.

Wolfson, O. & Sistla, A. P. & Xu, B. & Zhou, J. & Chamberlain, S. & Yesha, Y. & Rishe, N. (1999). Tracking Moving Objects Using Database Technology in DOMINO, In *Proc. of the Fourth International Workshop Next Generation Information Technologies and Systems (NGITS'99)*, Springer-Verlag LNCS, pp. 112-119.