On Real-time Distributed Geographical Database Systems*

Manhoi Choy[†**] Mei-Po Kwan[‡] Hong Va Leong[†***]

[†]Department of Computer Science
University of California at Santa Barbara
Santa Barbara, CA 93106

[‡]Department of Geography
University of California at Santa Barbara
Santa Barbara, CA 93106

Abstract

Advanced Traveler Information Systems (ATIS) under the *Intelligent Vehicle Highway Systems* (IVHS) context require efficient information retrieval and updating in a dynamic environment and at different geographical scales. Some problems in ATIS can be solved based on the functionalities provided by GIS systems. However, extra requirements such as real-time response are not readily met in existing GIS systems. We investigate the use of GIS-based systems for applications in ATIS and we propose a new system architecture based on existing GIS technology and distributed computing technology. Issues on data modeling, data representation, storage and retrieval, data aggregation, and parallel processing of on-line queries in the proposed GIS-based systems are discussed.

1 Introduction

Transportation planning has been turning away from the solutions of building highways and transit routes to changing people's travel choices and making more efficient use of existing facilities. With the recent research focus on Intelligent Vehicle Highway Systems (IVHS), it is imperative to utilize advanced information processing and communications technologies to achieve improvements in efficiency and safety. As one of its major components, Advanced Traveler Information Systems (ATIS) essentially aims at assisting drivers in trip planning and decision making on destination selection, departure time, route choice, congestion avoidance and navigation.

Specific application requirement of ATIS has been quite demanding. In order to provide traffic information useful for people's travel decision, accurate congestion prediction, enroute real-time traffic warning, and alternate routing suggestions are needed. These operations require real-time processing on large data set over a detailed transportation network. Geographic Information Systems (GIS), which allow efficient storage, retrieval and manipulation of spatial and aspatial objects, can provide a basis for ATIS.

*This paper appears in the 27th Hawaii International Conference on System Sciences, Geographical Information System minitrack.

**This research is supported in part by the National Science Foundation under grant number CCR-9223094.

***This research is supported in part by the National Science Foundation under grant number IRI-9117094.

Most ATIS research has operated on simplified street network [VMK+93]. GIS, on the other hand, provides a realistic representation of environment for querying and processing. Other information useful for traveler decisions can be integrated through geo-referencing. GIS is also highly flexible in manipulating spatial objects and distance according to rules, and different scenarios can be simulated to test the "what-if" cases. GIS operations can help to define individuals' spatial and temporal constraints of accessibility [GKG93]. The commonly used GIS data models, however, are not without problem. For example, the raster data model divides space into regularly shaped and sized pixels, whereas the topological data model subdivides space into irregularly shaped regions, links and nodes [Fra92]. None of them, however, represents traffic movement and interaction very well and the problem of connectivity is not taken into account.

Although some GIS packages such as TRANSCAD and the NETWORK commands in ARC/INFO implement transportation functions like routing algorithms in the topological data model, they are not without problems. First, the link-node structure is basically planar and would not distinguish an intersection with an overpass which does not cross at grade. This would induce problems for routing unless additional structure in the data model is added. Second, the topological model does not replicate how human perceive the street network. We usually do not think of the street network as segments of links with intersection, but more as the street as a whole. As such, the topological data model is not a natural navigable database. Research on the Ontario Standard Labeled Road Network by Noronha and Goodchild [NG92] aims to overcome the above problems. However, it does not deal with area objects that associated with the street network. Moreover, how to represent and process multi-level transportation networks for micro-macro spatial modeling is still a technical issue needed to be solved [Nye90]. IVHS applications require operations at both regional level and local level. Information may need to be transmitted between different levels of modeling. A more efficient data model that overcomes the above problems is needed.

Object-oriented data modeling as an alternative for spatial databases was discussed by Worboys [Wor90] and Herring [Her92]. Gahegan and Roberts [GR88] suggested an intelligent, object-oriented GIS, which is concerned with increasing the functionality and efficiency of the object-centered approach, and hence increasing the effectiveness of GIS as an aid to analysis and decision-making. However, none of the above has been applied in an ATIS context. In this paper, we propose the use of a database model based on object-oriented data modeling. It is more natural to treat street networks as different classes of objects that we perceive in the real world. An object-oriented data structure also allows the distinction of an intersection and an overpass due to their membership of different classes. For the micro-macro modeling problem, the use of an object-oriented model facilitates the introduction of new classes across different levels as well as the introduction of new functions for these classes. This may provide a better interface to users and enhance multi-level spatial modeling.

In view of the demanding computation for answering queries in ATIS, especially routing problems like the traveling salesman problem (TSP), parallel computing is regarded as one solution. Chang et. al. [CM93] presented a traffic network simulation model for real-time applications in ATIS. The proposed simulation model is implemented on a parallel computer for an efficient cost/performance ratio. Their model is implemented with a parallel data structure design and a parallel logic. Preliminary research results show that the running time varies with different levels of model complexities but the parallel simulation methodologies offer a promising alternative in implementing real-time ATIS applications. Furthermore, Imielinski and Badrinath [IB92] discussed the use of mobile computers in distributed systems. However, as far as we know, the use of mobile computers in the area of ATIS has not been investigated.

In ATIS, information provided to travelers may be affected by decisions made by others in the system. Interrelated decisions for pre-trip planners include the decisions by household members. For enroute travelers, decisions made by other drivers in the system would affect predicted traffic conditions. As a result, some form of consistency control is needed. Kaysi et. al. [KBAK93] suggested a consistency check in the system design. However, the consistency issue is not directly dealt with in the database design. In addition to the quality of traffic information provided to travelers, the assurance of privacy is also important and should be integrated in the design of the database.

This paper aims at developing a comprehensive GIS-based system to handle the data representation and data modeling problems for applications in ATIS. Spatial and temporal data aggregations are discussed. The technologies of parallel processing and mobile computers are used. Finally, concurrency control and privacy issues are incorporated into our system.

The design of our system is application-specific and is targeted on the ATIS users. Applications include congestion prediction, and routing for pre-trip planners, enroute travelers and emergency vehicles. Data are collected constantly and used for statistical purposes in research on travel behavior and patterns. These data can prove to be useful for planning purposes as well. In addition, information on locations such as tourist attractions, restaurants, and hospitals is geo-referenced in our GIS-based system. As a result, value-added information like yellow page information and tourist information is readily available to the users.

The main contributions of this paper include:

- the introduction of a new distributed system architecture for ATIS using existing advances in communication networks, database techniques, and distributed system design techniques,
- the presentation of new data models for the representation of information in ATIS (and other components in IVHS) capturing the object-oriented characteristics, the relational properties, and the temporal variations of data,
- the introduction of data shipping in processing local queries as a means to optimize response time and improve the overall performance of the system, as well as function shipping in reducing channel contention and communication overhead,
- the exploitation of a network of distributed computers (or a parallel computer) to solve complex problems such as TSP for better responsiveness,
- the incorporation of privacy protection for sensitive data.

The organization of this paper is as follows: In Section 2 we give an overview of our distributed system architecture. In Section 3, we analyze the characteristics of information to be stored in the system and design data models for this information. In Section 4, we discuss the types of queries that are served in our system and how they can be handled efficiently in our architecture using the data models developed in Section 3. In Section 5, we consider another important issue in our system, the problem of privacy protection on sensitive data. We conclude with a brief discussion in Section 6.

2 System Architecture

Figure 1 shows an overview of our system. A central site is installed with a set of workstations connected by a local area network. A distributed database is maintained among these workstations. The central site performs global queries from users. A set of base stations is distributed throughout the region served by the system. Each base station is installed with one or more computers, with some information stored in the central site replicated. The base stations are responsible for handling some of the global queries that can be solved with information available at the base stations and for receiving requests and transmitting responses to mobile users in the district covered by the base stations. Depending on the workload of base stations, they may be equipped with computing units of different computational power. When a base station is overloaded, requests are forwarded to the central site. It is assumed that the central site has enough computational power to handle requests forwarded from the base stations.

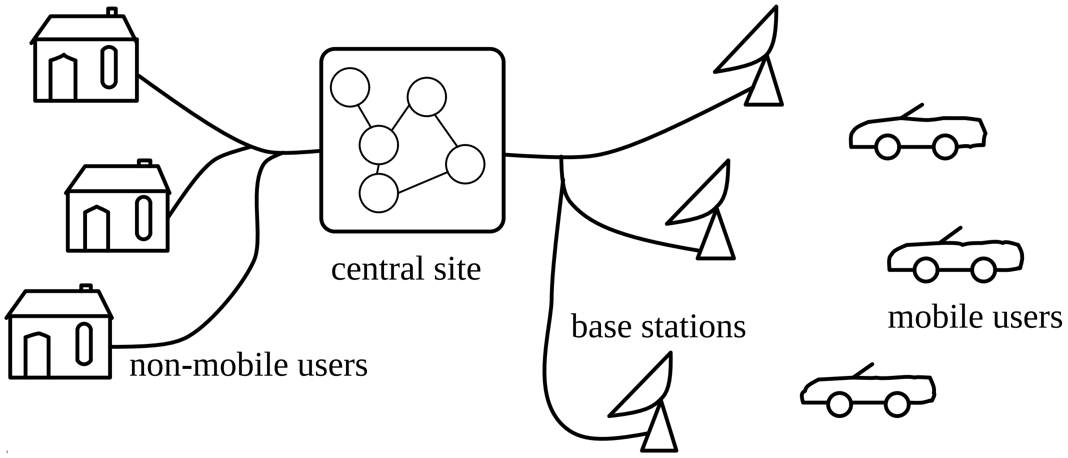

Figure 1: An Overview of the System

Since mobile users cannot be physically connected to a fixed station, communications between base stations and mobile users are based on a broadcasting medium, such as one using existing microwave broadcasting technology. This communication network can be superimposed on current cellular phone networks. On the other hand, non-mobile users communicate with the central site directly through telephone lines. Communications between base stations and the central site are based on another connection network. In small systems, this network may be embedded in the existing telephone system. However, in large systems, this network should be built on high bandwidth medium such as fiber-optics.

3 Data Modeling and Representation

Information in our system is classified as either *static* or *dynamic*. Information is classified as static if it remains unchanged over long periods of time and is classified as dynamic if it is updated frequently. For example, road maps, locations of stores, police stations, hospitals, etc., are classified as static information. Traffic conditions such as congestion levels, weather conditions, and the occurrence of accidents are classified as dynamic information. Different ways to maintain these two classes of information are needed in order to maximize the effectiveness of the system and to minimize the amount of data storage required. Two different data models and a combination of approaches for data storage are used here. For static information, a *relational object-oriented* model based on relational database techniques, attribute-list, and the idea of objects is designed while for dynamic information, a *temporal relational object-oriented* model based on a set of *temporal functions*, relational database techniques, attribute-list, and the idea of objects is developed. These are discussed separately in the following subsections.

3.1 Static Information and the Relational Object-oriented Model

In general, static information is initialized when the system is first setup and is rarely changed during the execution of the system (which may last for years). Even when there is a need to update the static information, it is assumed that relatively slow updating is tolerable.

Most of the information and requests that may appear in our system are location dependent. Furthermore, the relative positions between objects, the shape of objects, and a sense of locality are all important factors in processing requests. Consequently, we model the data in our system as a collection of objects sited on a multi-dimensional plane. Since data is considered as a collection of objects, the most natural approach is to use object-oriented modeling. However, if an object-oriented model is used in our system, the classification of objects by their relations usually requires the introduction of classes. The definitions of classes in object-oriented models are usually considered static and are not flexible in capturing complex and irregular relationships between objects. Moreover, in a pure object-oriented model, there is no systematic way to select and manipulate objects of a certain subclass.

On the other hand, in our *relational object-oriented* model, the way related objects are stored in the system is emphasized. In general, related objects are stored together so that they can be retrieved and updated more efficiently and proper indices are maintained among the related objects. Each object is characterized by a list of attributes. While the extent these attributes cover depends on the size of the system and the services the system is providing, the basic set of attributes should be enough to describe the appearance and the orientation of the objects. In the following subsections, we discuss the object-oriented aspect of our system and, in particular, the data representation of three different types of objects, namely point objects, line objects, and area objects. Then, we discuss the relational aspect of our system and the use of relational indices and data aggregation in our system.

3.1.1 Different Types of Objects

Point objects are those that are relatively small and do not extend to cover a significant area. Examples include buildings, police stations, hospitals, stores, and road-side radar detectors. A point object can be simply represented by its relative location and a list of attributes. Line objects include roads, railways, and rivers. Each of them can be represented by a sequence of line segments, which when connected together gives a good approximation of the object. Attributes are also associated with each line segment of the line objects. Area objects can be represented by a variety of methods. A small area object can be represented by a collection of point objects (note that under our definition, although point objects are small, they occupy a non-zero area). A regularly shaped object can be represented by a center point together with the size and shape of the object. A large or irregularly shaped object can be represented by a line object representing the boundary of the area object together with a direction bit, which is used to distinguish between the inside and the outside of the boundary. Moreover, these methods for the representation of area objects can be combined so that different parts of the object are represented using different methods.

3.1.2 Relational Indices and Information Aggregation

To maintain a sense of locality based on the relative locations of objects, information of the entire system is partitioned into *regions* and information within a region is partitioned into *districts*. For large systems, districts may also be further partitioned into sub-districts. For example, a region may represent an entire state, a district may represent a county, and a sub-district may represent a city. Regions are represented by their locations and the districts of which they are composed. Districts are represented by their relative location in the regions and the collection of objects in the districts. A number of coordinates are used to represent the locations of objects. The principal coordinates determine the region and the district in which an object appears and the remaining coordinates (called the relative coordinates) determine the relative location of the object in its district. Objects that are close to each other, i.e., within the same district (or sub-district if sub-districts are defined), are stored in closely related physical storages so that they can be retrieved and updated in a more efficient manner. Simple queries that only require object-wise information can be served by retrieving data directly from the corresponding district(s) and will not be discussed any more. More complicated queries may require filtering and processing of the information obtained from the districts. These ideas are discussed next.

Since the amount of data in the system is large, retrieving data from an entire district is costly. To enhance the performance of the system, we incorporate relational database techniques and the idea of information aggregation into our data model and data representation. Relational operations including *selection, projection, join, union*, and *difference* are supported and indices are built to handle commonly raised queries. These operations and indices allow efficient retrieval of related objects and are used to filter out unrelated objects. As an example, consider the following heuristic to approximate the shortest path between two locations in two different districts. We select and extract (by using the standard relational operation *selection*) only an outline of the districts along the straight line connecting the two locations. This outline contains main freeways (and highways) in the districts. The retrieval of this outline is performed with the help of an index that links together all the main freeways in the districts. When the appropriate freeway (or highway) entrances and exits are identified, a shortest path can be computed based on the outline. However, some main streets may also be considered in the outline if their inclusion would lead to better connectivity. The criteria under which main streets should also be considered may be either properly setup initially and manually updated periodically or learnt during the execution of the system. In the latter case, main streets are included in the road map extractions based on dynamic information stored in the system. In general, if the system finds (from statistics) that a certain main street is being frequently used by users as a connection between freeways, the main street is more likely to be included in the district's road map outline. The storage ofathis stlmistical information will be discussed in Section 3.2.

To further reduce the amount of information needed to be retrieved, information should be aggregated. As an example for the aggregation of static information, consider the problem of retrieving population information in a certain region. Instead of returning the population of every location representable in the system, the population of larger area units is returned. In our system, the population of the larger area units is obtained by summing up the population of the respective smaller area units.

3.2 Dynamic Information and the Temporal Relational Object-oriented Model

Dynamic information is collected, analyzed, and stored in the system during normal execution. New information is input frequently and the demand on the availability of the most up-to-date information may be bursty. As new information is frequently input, the problem of how to maintain outdated information arises. Since useful information may be deduced from old data, this outdated information should not be discarded completely. To minimize the amount of data storage required and yet to be able to retrieve important information, some form of information aggregation is needed. Two dimensions of data aggregation, namely *spatial* and *temporal* data aggregation, are needed in our system. Spatial data aggregation minimizes the data storage needed for spatially distributed information and reduces the amount of data communications needed for many common queries. Temporal data aggregation minimizes the data storage needed for outdated information and reduce the computational requirements for many common queries. These two kinds of data aggregation are elaborated next.

To illustrate the idea of *spatial* data aggregation of dynamic information, consider the problem of representing congestion levels in the freeway systems. Not every location on freeways representable in our system has an associated value of the congestion level. Instead, congestion levels are sampled at specific locations and averaged over small areas in our system. The congestion level of individual location is obtained by interpolation from congestion levels at nearby locations (along the connecting freeways). Consequently, queries requesting for this information (such as the query requesting for the graphical display of the congestion levels of a certain district) can be served with a smaller amount of data communication. A wide variety of point based interpolation methods such as moving average, Fourier analysis, and stochastic [Lam93] exist. For the purpose of interpolating congestion levels, which is the best method is still an open question.

To illustrate the idea of *temporal* data aggregation of dynamic information, consider the problem of the processing of a query from a pre-trip planner. A pre-trip planner from Santa Barbara visiting Las Vegas by car during the coming weekend would like to find the least congested route. To predict the traffic condition on the way to Las Vegas during the coming weekend, the system needs to know the traffic condition on previous weekends. With temporal data aggregation, only traffic condition averaged per hourly intervals are stored instead of every piece of information that was available in the previous weekends. The length of intervals that averages are taken may vary depending on the fluctuation of traffic condition. For example, for freeways near cities shorter intervals should be used and for freeways in the deserts longer intervals suffice. Thus, temporal data aggregation limits the amount of outdated data remaining in the system and reduces the amount of computation needed for the request.

Temporal data aggregation is in general much easier to automate compared with spatial data aggregation because of the former's *linear* characteristic. In our system, we attempt to incorporate temporal data aggregation systematically and automatically. Given any dynamic information unit, a set of *temporal functions* is provided to retrieve the past history of the unit. The past history of an information unit may be its statistics, or some averaged value of the unit in a specific time intervals. If prediction based on past history is possible for the information unit, retrieving *future* (or predicted) value of the unit may also be provided. Depending on the data storage capacity of the system, there may not be enough information stored to return the past history of some information units and, in that case, an appropriate error message is returned. Furthermore, the retrieval of information units based on a set of temporal functions can be extended to a more general fashion. In fact, every query can be composed with any of the temporal functions to yield a new query. For example, for the pre-trip planner visiting Las Vegas, he/she may issue a query to retrieve the related traffic condition maps during the coming weekend as predicted by the system.

Incorporating temporal relations in the database does not have to be restricted to dynamic information. It is also possible to model all the information in our system with a temporal relational object-oriented model. However, since static information is less likely to be changed over very long period of time, we do not intend to use such a model for static information. Instead, a log keeping the updates of the static information should be sufficient. For example, if a user wants to find out the map of a district five years ago, the user is required to go through the updates kept in the log.

4 Queries and Updates

ATIS should be able to handle user queries from various origins and destinations and at different scales. We classify queries according to the type and amount of information to be retrieved from the database into local queries and global queries. Local queries can be handled efficiently by retrieving a small amount of information and performing some processing on local computers. Global queries need to access larger amount of information (or aggregated information not available locally), which may be partitioned among several base stations. Base stations are usually mapped to the districts they are located. Global queries may be processed in a base station, or forwarded to the central site and processed in parallel.

We propose a hierarchical information caching scheme to minimize the data transmission time and access latency. With a network of distributed computers and abundant storage, the central site maintains all GIS information. Each base station caches the information about its district and aggregated information about neighboring districts. Mobile computers cache only local information. Through caching, information retrieval time is greatly curtailed. In the same vein as hierarchical information caching, we envision a hierarchical query processing structure. Local queries are solved on local computers. Global queries are solved by the computing facilities at the base stations or the central site. Base stations forward queries to the central site if they do not have enough information to handle the queries or they are overloaded. In this manner, the workload is distributed among all possible processing agents in the system, creating a higher throughput.

In addition to query processing, the system must also handle the updating of information. Static information, as we mentioned earlier, is only updated infrequently and relatively slow updating is tolerable. However, dynamic information is updated more frequently and may affect the processing of queries. The results of the queries must observe the effects of updates in a consistent manner. This brings in the necessity of concurrency control.

4.1 Local Queries

Local queries involve only local information. A very useful piece of local information is the local map of a district, which can be used to guide the driver through local streets and to search for local facilities such as shopping malls, restaurants and scenic attractions. Other useful information may include indices and details of some of these facilities. Our focus will be on how to handle the data requirement for these queries, whereas the algorithms are assumed to be off-the-shelf and readily available.

For non-mobile users, local information is obtained from the central site. For mobile users, in order to process queries locally, efficient ways of dispersing local information are needed. Due to the fact that transmission channels are scarce resources, the simple mechanism of requesting the information on a demand basis from the base station to each user is not only inefficient, but also impossible. This can be verified by a simple calculation on the channel capacity and size of the maps. We use the efficient broadcast mechanism inherent in a base station to transmit the local maps periodically to every user in the district. To further save the communication bandwidth, local maps are transmitted in a compressed form. Any standard compression technique suffices. Each user will receive a compressed local map within a short period when the user enters a new district. The local map received is uncompressed, cached and used. This reduces the access latency and, more importantly, channel contention.

4.2 Global Queries

Global queries need information on the whole region and is not possible to solve locally. Therefore global queries are forwarded to base stations. Medium scale global information and aggregated information are replicated at each base station. This allows base stations to be capable of solving most global queries. This is a function shipping strategy for global queries rather than data shipping, the strategy adopted in most local queries. We make this design decision to reduce the amount of information to be transmitted over scarce channels to the user computers. We are more concerned with design decisions such as the data representation and shipping than the actual algorithm to solve the queries. In this section, we present treatments to the shortest path problem, emergency vehicle routing and the TSP (traveling salesman problem) under our framework.

The shortest path problem is one of the most standard problems in graph theory and there are numerous solutions implemented on various GIS. The solution strategy mentioned earlier in Section 3.1.2 requires the use of outlined information of freeway systems and main streets. Static information of maps, capacity of the individual freeway and freeway system outline are readily available from the databases of the base stations. The dynamic information of traffic congestion includes, for example, traffic volume on each section of the freeways, an exponentially weighed moving average of the number of arriving and departing vehicles from the freeway system, and the effects of particular events, such as the location and number of lanes affected by a road construction or a traffic accident. Base station databases store all information about the local district as well as aggregated information about other districts.

Emergency vehicle routing must be made in reaction to accident and emergency. The system should provide guidelines to allow fast dispatch of service vehicles to the scene of emergency. The system determines, on the basis of the location and type of service vehicles, the vehicles that are capable of carrying out the mission and provides the fastest routes for them. These global queries require further dynamic information such as a profile of the police and emergency vehicles in the district. In collaboration, the system should issue warning messages to other vehicles, aiming at creating even faster routes for the service vehicles.

The problem of finding a guideline to tour around a city subject to a list of requirements is a generalization of the TSP. The TSP is an extremely difficult problem (NP-complete) [GJ79] to be solved efficiently. Heuristics are usually adopted to generate solutions that are close to optimal in a reasonable amount of time. Whether an exact algorithm or a heuristic is to be used, a nice feature of this problem is that it can be divided into subproblems which are quite independent of one another. This leads to a very effective utilization of the distributed computers in the central site, by solving the subproblems on different processors. In fact, the speedup is almost linear to the number of processors used to solve the problem. Problems of this nature justify the use of network of inexpensive computers for a cost effective solution. Better load balancing is achieved with a network of computers when many small global queries are processed.

4.3 Updates and Concurrency Control

As mentioned earlier in Section 3.1, updates to static information occur rather infrequently, and we are concerned mainly with updates to dynamic information. There are various sources of dynamic information. Data collected by various sensor devices located on freeways, information provided by vehicles, information provided by helicopters monitoring traffic flow, and accident reports are examples of sources of dynamic information. Due to the large volume of dynamic information, an efficient way to update information in the system is needed. Also, due to the concurrent updating of dynamic information and user querying, concurrency control is essential. We describe below the way information is updated in our system and how concurrency control is achieved.

Dynamic information is sent between base stations and the central site in one of two modes. Under *normal* operation mode, information from base stations is exchanged through the central site periodically. This allows base stations to have an updated view of other base stations. Under *emergency* operation mode, information is exchanged between base stations through the central site immediately. This allows critical information to be available in other base stations as soon as possible. For example, in case a traffic accident occurs in one district, new traffic flow information is immediately transmitted to other base stations. Based on the new traffic flow information, the other base stations may discourage traffic flow towards the district at which the accident occurs and suggest alternative routes.

Next, we consider the issue of concurrency control. The problem of consistency arises when the database is updated in the midst of the processing of a query. Static information updates can be made to appear consistent in an atomic manner with respect to the processing of queries and other updates to the system, by enforcing the rules of concurrency control schemes. This avoids bringing the system to a standstill when a large amount of static information needs to be updated in a consistent manner.

For dynamic information updates, a simple scenario suffices to illustrate the problem of consistency. Consider the occurrence of a traffic accident on a freeway. This information is recorded in the local base station and propagated to the central site and neighboring base stations. Owing to the sudden slowing down of traffic, a lot of drivers may query for an alternative route. Suppose routes α and β are available. When such queries flood the base station, route α is a better alternative based on the current situation. Without any concurrency control scheme, the answers to all such queries will be α and the net result is that most, if not all, of the vehicles are diverted to route α, creating another congestion. Each answer of α may affect the traffic flow by guiding one more vehicle to route α. In terms of concurrency control theory in multi-user database systems, we should model the query of alternative route as an *update transaction*. The notion of a transaction guarantees that each query will see the effect of another. Even though the queries are executed in parallel, the database behaves *as if* the queries are executed one after another. Sequential execution of queries guarantee correctness. In the above example, after a certain number of queries are executed, the best alternative route may become β.

A commonly used correctness condition for a multi-user database is the *serializability* of transactions [Pap79, BHG87]. This calls for the property of *as if* sequential execution of transactions. For most updates in an ATIS database, serializability can be enforced by either the *two phase locking* rule [EGLT76], or the *time stamp ordering* rule [Ree78]. In the context of the above query/update situation, where we do not require an absolutely shortest alternative route, it is possible to relax the correctness condition of serializability to *bounded ignorance* [KB91] or *bounded inconsistency* [WA92]. This allows a greater degree of concurrency and better performance. A variation of such relaxed correctness condition is used in our system. We would like to bound the error E_i in the estimated routing time for each transaction i (or query in our system). This error is defined as the difference between the routing time obtained if the transaction is executed alone and the routing time obtained if the transaction is executed concurrently with other transactions. If the sum of the errors of all the currently active transactions is smaller than a given bound, then the transactions are allowed to execute. Otherwise, the transactions may be blocked or an alternate route may be chosen stochastically among all but the best alternate routes. Generation of a suboptimal answer as well as the termination of a transaction, bring down the sum of the errors. As the sum of the errors is lowered, more transactions can be executed within the error bound.

5 Privacy Protection

Privacy and security are important issues in a public information system. When base stations have access to the information associated with each user's vehicle, the unrestricted disclosure of this information constitutes a violation to the Act of Privacy. As an illustration, consider a base station that knows the location of a vehicle every now and then. By taking the temporal and spatial information of the vehicle, it may well reveal that the vehicle is traveling at a speed of 70 mph (exceeding the 55 mph limit). Even though the driver of the vehicle is violating the traffic law, he will not want this to be disclosed, if he is not caught on scene by the police. Similarly, the whereabout of a vehicle may be a piece of sensitive information, especially when the whereabout is clearly identified with a time. The traffic pattern of a vehicle is regarded as private information. Mechanisms must be devised to ensure privacy protection. Also, sufficient information must be available to the system, to answer queries.

An easy solution to this is the *anonymous vehicle policy*. The system preserves information about each vehicle, but removes the identification of the vehicle. This anonymous information can then be extracted to generate the aggregated information. Vehicle identifications are only kept during the query computation and destroyed immediately afterwards, leaving only enough information for statistical purposes. To satisfy billing purposes, the originating times of queries and resources spent on their processing must also be recorded. This information must be decoupled from the content of the query and the location from which the query is originated to ensure privacy. For example, information can be decoupled into two relations $statistics = \langle vehicle, time, query\_type, query\_expense\rangle$ and $private = \langle vehicle\_id, location, query\_content\rangle$. The decoupling of information in such a manner is a feasible, but not the best, solution as it becomes impossible resolving certain disputes between the users and the service provider. Also, it is not possible for users to present records of their vehicles as evidence of not involving in some criminal acts.

The technique of cryptography can be used to maintain full accounting information without compromising privacy. We propose to employ a public key encryption scheme, such as the RSA scheme [RSA78]. There are distinct functions associated with each customer: an encryption function $E(P)$ and a decryption function $D(P)$ for a piece of information P. The pair of functions must satisfy $D(E(P)) = E(D(P)) = P$ for any P. It must be impossible to deduce D from E and for any P' very similar to P, $E(P)$ and $E(P')$ must look very different. The function E is known to the public and D is kept secret by the user. The tuples of *statistics* and *private* are indexed by sequence numbers. A relation *connection* is defined to relate the records of *statistics* and the records of *private*. A pair (a, b) is in *connection* if a is the sequence number of a record in *statistics*, b is the sequence number of a record in *private*, and these records are resulted from the same query. Privacy protection of sensitive information is achieved by protecting the disclosure of the relation *connection*. Instead of storing the relation *connection* directly, it is stored as encrypted tuples of the form $\langle s.seq, E(p.seq \cdot D(s.seq))\rangle$ where $s.seq$ and $p.seq$ are the sequence number of the corresponding *statistics* and *private* records. A query is executed as follows: the system generates $s.seq$, the user computes $D(s.seq)$ and returns to the system. The system verifies that the user is not lying when $E(D(s.seq)) = s.seq$. When the query completes, the tuple $\langle s.seq, E(p.seq \cdot D(s.seq))\rangle$ is generated and stored. It is impossible for the system to fake the tuple, since $D(s.seq)$ cannot be generated without D. It is also impossible for the system to deduce $p.seq$ from $s.seq$ without knowing D. The user can disclose the information by presenting his key D to compute $D(E(p.seq \cdot D(s.seq))) = p.seq \cdot D(s.seq)$ and to retrieve the complete record.

Despite the cryptographic approach, the complete system would gather accounting information in an anonymous manner and just maintain the encrypted copies for dispute resolution and other mutually consent usages. Different levels of security and privacy can be defined based on the sensitivity of the data, and different encryption techniques can be employed.

6 Conclusion

In this paper, we have proposed a system architecture composing of a central site, a set of base stations, non-mobile computers hooked up to the system via phone lines or fiber optics, and mobile computers connected to the system via a broadcasting medium. Three levels of processing powers, from users' personal computers, to base stations' workstations, to the central site's distributed computers contribute to a high degree of concurrency in the system. New data models, the *relational object-oriented* model and the *temporal relational object-oriented* model, are introduced to represent IVHS information. These data models allow efficient representation of both static and dynamic information that are inherent to IVHS. Information aggregation is supported in both spatial and temporal domains.

Although our system is mainly designed for ATIS users, it can be extended to include many other functions. In the extreme, it can be extended to a general database system that provide general information to the public. More work is needed to investigate the feasibility and the cost effectiveness of such a system. Future work includes the investigation of implementation issues of our system. Also, simulations on the run-time behavior of the system including user query pattern, channel utilization, response time and data storage requirement should be examined. Applications of the newly developed data models in other systems should be studied.

References

[BHG87] P. A. Bernstein, V. Hadzilacos, and N. Goodman. *Concurrency Control and Recovery in Database Systems.* Addison Wesley, Reading, Massachusetts, 1987.

[CM93] G. Chang, T. Junchaya, and A. J. Santiago. A real-time network traffic simulation model for ATMS applications. In *Transportation Research Board 72nd Annual Meeting, January 10-14*, 1993.

[EGLT76] K. P. Eswaran, J. N. Gray, R. A. Lorie, and I. L. Traiger. The Notions of Consistency and Predicate Locks in a Database System. *Communications of the ACM*, 19(11):624–633, November 1976.

[Fra92] A. U. Frank. Spatial concepts, geometric data models, and geometric data structures. *Computers and Geosciences*, 18(4):409–418, 1992.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, New York, 1979.

[GKG93] R. G. Golledge, M.-P. Kwan, and T. Garling. Computational-process modelling of household travel decisions using a geographic information system. *Papers for regional science.* To appear.

[GR88] M. N. Gahegan and S. A. Roberts. An intelligent, object-oriented geographical information system. *International Journal of Geographical Information systems*, 2:101–110, 1988.

[Her92] J. Herring. TIGRIS: a data model for an object-oriented geographic information system. *Computers and Geosciences*, 18(4):443–452, 1992.

[IB92] T. Imielinski and B.R. Badrinath. Querying in highly mobile distributed environment. In *Proceedings of the 18th International Conference on Very Large Databases*, pages 41–52, 1992.

[KB91] N. Krishnakumar and A. J. Bernstein. Bounded Ignorance in Replicated Systems. In *Proceedings of the 10th ACM Symposium on Principles of Database Systems*, pages 63–74, May 1991.

[KBAK93] I. Kaysi, M. Ben-Akiva, and H. Koutsopoulos. An integrated approach to vehicle routing and congestion prediction for real-time driver guidance. In *Transportation Research Board 72nd Annual Meeting, January 10-14*, 1993.

[Lam93] N. Lam. Spatial Interpolation Methods: A review. *The American Cartographer*, 10(2):129–149, 1993.

[NG92] V. T. Noronha and M. F. Goodchild. The Ontario Standard Labeled Road Network: integration of emergency services and other user needs into a conceptual data model. In *Proceedings, Fourth Canadian Conference on GIS*, Ottawa, March, 1992.

[Nye90] T. Nyerges. Locational referencing and highway segmentation in a geographic information system. *ITE Journal*, March, 1990.

[Pap79] C. H. Papadimitriou. The Serializability of Concurrent Database Updates. *Journal of the ACM*, 26(4):631–653, October 1979.

[Ree78] D. P. Reed. Naming and Synchronization in a Decentralized Computer System. Technical Report MIT-LCS-TR-205, Massachusetts Institute of Technology, Cambridge, Massachusetts, September 1978.

[RSA78] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21:120–126, Feb 1978.

[VMK+93] K.M. Vaughn, A. A. Mohamed, P. P. Kitamura, R. amd Jovanis, H. Yang, N. E. A. Kroll, R. B. Post, and B. Oppy. Experimental analysis and modelling of sequential route choice under ATIS in a simplistic network. In *Transportation Research Board 72nd Annual Meeting, January 10-14*, 1993.

[WA92] M. H. Wong and D. Agrawal. Tolerating Bounded Inconsistency for Increasing Concurrency in Database Systems. In *Proceedings of the ACM SIGMOD-SIGACT Symposium on Principles of Database Systems*, pages 236–245, June 1992.

[Wor90] M. F. Worboys, H. M. Hearnshaw, and D. J. Maguire. Object-oriented data modelling for spatial databases. *International Journal of Geographical Information Systems*, 4(4):369–383, 1990.