Title

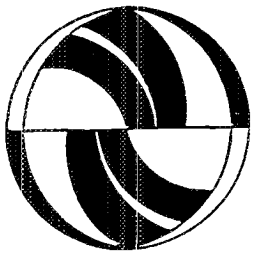
A Review of Object-Oriented Approaches in Geographical Information Systems for Transportation Modeling

Authors

Kwan, Mei-Po
Golledge, Reginald G.
Speigle, Jon M.

Publication Date

1996



A Review of Object-Oriented Approaches in Geographic Information Systems for Transportation Modeling

Mei-Po Kwan
Reginald G. Golledge
Jon M. Speigle

Reprint
UCTC No. 412

The University of California
Transportation Center

University of California
Berkeley, CA 94720

The University of California Transportation Center

The University of California Transportation Center (UCTC) is one of ten regional units mandated by Congress and established in Fall 1988 to support research, education, and training in surface transportation. The UC Center serves federal Region IX and is supported by matching grants from the U.S. Department of Transportation, the California Department of Transportation (Caltrans), and the University.

Based on the Berkeley Campus, UCTC draws upon existing capabilities and resources of the Institutes of Transportation Studies at Berkeley, Davis, Irvine, and Los Angeles; the Institute of Urban and Regional Development at Berkeley; and several academic departments at the Berkeley, Davis, Irvine, and Los Angeles campuses. Faculty and students on other University of California campuses may participate in Center activities. Researchers at other universities within the region also have opportunities to collaborate with UC faculty on selected studies.

UCTC's educational and research programs are focused on strategic planning for improving metropolitan accessibility, with emphasis on the special conditions in Region IX. Particular attention is directed to strategies for using transportation as an instrument of economic development, while also accommodating to the region's persistent expansion and while maintaining and enhancing the quality of life there.

The Center distributes reports on its research in working papers, monographs, and in reprints of published articles. It also publishes *Access*, a magazine presenting summaries of selected studies. For a list of publications in print, write to the address below.

University of California Transportation Center

108 Naval Architecture Building
Berkeley, California 94720
Tel: 510/643-7378
FAX: 510/643-5456

A Review of Object-Oriented Approaches in Geographical Information Systems for Transportation Modeling

Mei-Po Kwan
Department of Geography
Ohio State University
Columbus, OH 43210-1361

Reginald G. Golledge
Department of Geography
and
Research Unit in Spatial Cognition and Choice
University of California, Santa Barbara
Santa Barbara, CA 93106

Jon M. Speigle
Department of Psychology
University of California, Santa Barbara
Santa Barbara, CA 93106

Reprinted from
Santa Barbara Geographical Press
Submitted to *Transportation Research, Part C: Emerging Technologies*
(1996)

UCTC No. 412

The University of California Transportation Center
University of California at Berkeley

Abstract:

The objective of this paper is to review object-oriented (OO) approaches to data modeling and data handling and their usefulness in transportation planning and modeling in general and Intelligent Transportation Systems in particular. The paper begins with a discussion of the current GIS data model for representing a transportation network and the most common database management systems used in the context of transportation planning. We then discuss object-orientation, its different properties and its usefulness in representing a multi-modal, multi-scale network with hierarchical road types. We also discuss alternative database management schemes. Finally we review some existing systems and discuss the implications of adopting an object-oriented perspective.

Purpose

Intelligent Transportation Systems (ITS) utilize advanced communication and transportation technologies to achieve traffic efficiency and safety. The different components of ITS include Advanced Traveler Information Systems (ATIS), Automated Highway Systems (AHS), Advanced Traffic Management Systems (ATMS), Advanced Vehicle Control Systems (AVCS) and Advanced Public Transportation Systems (APTS) (Figure 1). Development of a system for ITS critically depends on our capability to incorporate a vast amount of information about the locations of features and to maintain a complex representation of the transportation network for various activities within a geographic database (Watling and Vuren, 1993; Schofer, Khattak, and Koppelman, 1993; Kwan, 1994; Kanninen, 1996;). The system therefore needs to be constructed based on an integrated and comprehensive Geographic Information System (GIS) (Worboys, 1992, 1994; Tomlin, 1990; van Oosterom and van den Bos, 1989; Wiegand and Adams, 1994). Unlike the simplified planar graph theoretic, transport network representations used by current ITS, GIS are able to provide a more realistic representation of elements of the complex transportation environment. As such, there is considerable potential for GIS to become prominent for constructing a real-time multi-strategy travel decision support system as we conceive it.

[Insert Figure 1 about here]

The first ITS requirement which is met by a GIS is the capability to represent the transportation network in detail sufficient to perform different network algorithms, modeling and simulations. In the real world, a transportation network has different types of roads, (e.g. highway, arterials, local streets, etc.) and different types of intersections (e.g. signalized and non-signalized) that are of interest to ITS builders. For some applications, information on intersections, lanes and lane changes such as turn lanes and

merging lanes, highway entrances and exits, etc., is also important. Other applications may require geometric representation of road curvature and incline.

Current GIS data models usually represent a network as a collection of links with nodes created at the link intersections. Unless additional structure is added, the link-node nature of the current commercially available topological data model is basically planar and cannot distinguish an intersection at grade from an intersection with an overpass or underpass which does not cross at grade. The difficulty in accurately representing overpasses or underpasses may lead to problems when running various routing algorithms (e.g. recommending that a traveler make a left-turn at an intersection that proves to be an overpass!).

There are other situations in which we may want to differentiate different types of nodes. In a link-node structure, a node is created when there is an intersection. However, when the traveler needs to make a decision to move along the network, there are many decision points that they need to encounter in addition to intersections (like a barrier along a street). Sometimes, different nodes may lead to different behaviors. For example, we usually make a U-turn at a dead-end node or a barrier. Currently existing software does not handle this differentiation. An object-oriented approach is thus promising in this aspect.

A second ITS requirement is that, since ITS applications have to operate at various spatial scales, a multi-level representation of a transportation network is needed. At the local level (i.e., city or county), the transportation network needs to be represented in very great detail for navigational purposes, whereas only the major elements such as the interstate or state highways need be represented for travel at the regional, state or interstate levels. Further, relationships between data at different scales have to be established. Only through the use of a more efficient geographic data model can we hope to overcome these and other related problems (van Oosterom and van den Bos, 1989; Roberts, Gahegan, Hogg, & Hoyle, 1991). Several recent attempts to tackle such

problems within transportation modeling have focused on the object-oriented approach (e.g. Mainguenaud, 1995; van Oosterom and Schenkelaars, 1995).

A third requirement is that ITS must handle the real-time updating of the traffic patterns and transmission of information . To provide travelers with timely decision support, an Advanced Traveler Information System (ATIS) needs to receive and transmit data in real-time through a communications network which connects a vast number of system elements, including traffic sensors and location-tracking devices. Even for a mid-sized city like Santa Barbara (approximately 180,000 people and 13,000 segments in its transport network), the real-time update and transmission of data is already a very challenging problem. To keep information on the dynamic environment current and self-consistent even in the presence of concurrent updates and queries from around the network, we need to go beyond the existing data handling technologies of current ITS and GIS. The system will likely be distributed and will require spatial partitioning tailored for effective access.

To meet these requirements, we explore here the use of object-oriented data modeling and database technologies. Below, we review the object-oriented approach in relation to transportation planning and modeling, examine and compare object-oriented systems with others, and discuss the implications of adopting an object-oriented GIS.

Object-oriented approach - An overview

General

Object-orientation (OO) is a technology that integrates various programming and modeling techniques for analyzing the world. An object-oriented model is a collection of discrete objects, their associations and their interrelationships. An object-oriented model also is an enumeration of the ways in which a system transforms its values and an

elaboration of the timing, sequencing and control of events. It is a methodology which formalizes many of the operations performed in any modeling exercise.

Object-oriented modeling has been reviewed and discussed extensively (e.g., Rumbaugh, Blaha, Premerlani, and Lorensen, 1991; Martin, 1993; and Booch, 1994) as well as in several texts specific to database systems (Khoshafian & Abnous, 1990; Kim, 1990; Gupta and Horowitz, 1991; Khoshafian, 1993; Loomis, 1995). The major concepts are well-documented in these books and reports. Major constructs of the OO approach, including object identity, encapsulation, inheritance, generalization, specialization, aggregation, and polymorphism, are discussed also in this literature and in an increasing number of GIS related journal articles (e.g. Worboys, 1992, 1994b; Worboys, Hearnshaw, & Maguire, 1990; Egenhofer and Frank, 1989). We summarize this material below. Although we will use OO terminology, some of the concepts predate the development of object oriented methods while others are particular to it.

Object identity

Object-oriented models look at the real world in terms of tangible objects and how they behave. For example, the transportation network can be represented as different objects such as highways, intersections, and ramps. Each object has its own *methods* which specify the different manipulations possible by an object of that type. A *message* or *request* sent to an object will invoke the object's methods. For example, calculating the shortest travel time between two locations may be represented as a method of a *street network* object.

Note that existing GIS usually represent the transportation network in terms of links (i.e. road segments) with nodes (i.e. street intersections). This is called the link-node structure. Each line segment typically has a unique identification. A relational database such as ARCINFO may be used to link the spatial objects with other attributes. In such a database, each record is distinguished by the data values it contains. Its identity is not

easily retained if some of its properties change over time (Korth and Silberschatz, 1991).In the object-oriented system, each object corresponds to an entity in the modeled world. Its identity is retained even if its properties change over time. For example, if an object migrates several stages (i.e. change as the transportation network changes), it will still keep the unique identification through *versioning*. As such, we say that the object identity (OID) is *persistent*. We can use versioning to represent the changes of, for example, census tracts over time (Worboys, 1992).

Classification

Objects can be classified into different object types. A type description generally consists of a description of the attributes and methods supported by objects of that type. A type may also have additional attributes which are not associated with any individual object but with the type as a whole. The type description is also referred to as defining the *interface* of the type. An object class is a software implementation of an object type. A type may be represented by several classes. Classification in object models is an abstraction method and models the real world in terms of a hierarchical taxonomic structure. Basically all of the following OO concepts concern aspects of classification.

In the object model, a class can be further divided into subclasses. Elsewhere, we have experimented with a system that consists of a spatial object hierarchy which examines the network as a hierarchy of point, polyline, polygon and polyline list objects (Kwan, Speigle and Golledge, 1996). Figure 2 details the conceptualization of a hierarchy for the subclasses *Node* and *Link* to represent the transportation network.In Figure 2 we graphically depict a subset of the basic elements of a vector-based, transportation model. We use a subset of the notation of Rumbaugh and colleagues' Object Modeling Technique (OMT) (1991). Each box represents a different class and has three sections: the class name, class attributes, and class methods. Attributes are indicated either graphically or by listing their labels and types in the section below the

class name. The lines connecting different classes represent relationships between those classes. The type of the relationship is indicated by the symbols at the ends of the attaching lines. Inheritance is indicated by a triangle connecting the superclass to its subclass. Aggregation is indicated by a diamond on the side of the class which contains the aggregated attributes. These relations are labelled with the name of the aggregated attribute. A one-to-one relationship is indicated by an unadorned line; a zero-to-many relationship is indicated by a circle.

The *Point* and *Polyline* classes are derived from the base class *Spatial*. Spatial defines the interface supported by all derived classes. These methods are *abstract* because they are not actually supplied by Spatial, but are supplied by the derived class. The class *Node* is derived from Point; and class *Link*, from Polyline. With the exception of "turn costs" for Node, the attributes are indicated graphically. A Link includes references to a pair of Nodes, in addition to the list of intervening vertices inherited from Polyline. We can further model the real world Node as *Stop* and *No-stop* Node subclasses as in the real world transportation network (Figure 3). The Link class may be further modeled as *Highway*, *Interstate* and *Addressed* road types (Figure 4).

[insert Figure 2 about here]

[insert Figure 3 about here]

[insert Figure 4 about here]

Generalization

Several classes of objects, which have some properties and operations in common, can be grouped by forming a more general super class. Each subclass would describe a specialization of the super class. This concept is especially important for the abstraction

of a real transportation network. For example, the object type Highway, Interstate and Addressed can be grouped to a super class *Road* (Figure 4).

Association

A relationship between similar objects can be grouped together to form a higher level object (Brodie, 1984). The term *set* is used to describe the association, and each associated object in the set is called a *member* (Egenhofer and Frank, 1989). This is important to GIS because spatial relationships may be modeled as associations. In the transportation context, an association can represent the relationships between network elements and other spatial objects. For example, a highway may be inside a city and together this forms a set (highway in city). An ordered collection of intersections can be grouped to form a road and this is called an ordered association.

Aggregation

Several objects can be grouped together to form a composite object, creating a higher-level object which may itself be referred to. The model of this process is called aggregation. For example, the object type *Highway* is an aggregation of lanes, road sign positions, exit entrance and ramps, and so on.

Details of objects are suppressed in the constituent or aggregated objects. Every instance of an aggregate object can be decomposed into instances of the component objects while keeping its own functionality. However, the operations on the aggregate may not be compatible with operations on its components (Egenhofer and Frank, 1989).

Inheritance

Inheritance is a major-type/sub-type relationship. It is intrinsically related to generalization. Inheritance is the mechanism by which a type inherits properties (data structure and methods) of its super type. A type Road may represent the generalization of arterial, collector and highway. The arterial, collector and highway may be considered specializations of Road. The supertype/subtype relation is indicated in the

subtype definition. The subtype is said to be *derived from* or to be the *child of* the *parent* type. For example, a highway derived from Road would inherit the properties of Road (Figure 4 and 5). By being a subtype of Road, it has the attributes of length, number of lanes, speed limit, direction and divider type. Inheritance is one of the most powerful and distinctive characteristics of object orientation. The act of inheriting properties from more than one super type is called *multiple inheritance*.

[insert Figure 5 about here]

Polymorphism

Polymorphism generally represents the quality or state of an object, which is able to assume different forms. In a programming context, polymorphism is the mechanism by which the same sequence of programming statements can be used to manipulate objects of different types. One form of polymorphism is called overloading. With overloading, the same operator (e.g. the "addition",+, operator or the "print yourself" message) can perform different functions depending on the class of the object receiving the message . It is a very important feature for transportation modeling. For example, the link-node structure assumes that the transportation network is represented as links and nodes, and there is no differentiation between these links and nodes. The delay function in this type of system cannot be easily modeled differently for signaled intersections versus non-signaled intersections, or for highways versus local streets. Polymorphism within an object-oriented context allows these different network elements to be modeled using different functions, but with the same, overloaded operator representing the delay function. Khoshafian and Abnous (1990) listed the advantages of overloading in terms of programming: (1) extensibility - the same algorithm or function may be applied to instances of many classes (some of which may be defined in the future) without modifying its code; (2) development of more compact code because there is no need to

list out all the situations for different classes; (3) clarity - the code is more readable and comprehensible. A penalty is paid for the run-time binding of messages to functions which depend on an object's type. . Nevertheless, specially designed constructs such as hash tables and indexes have been developed to lessen the performance penalty of dynamic binding (Khoshafian and Abnous, 1990).

In some languages, parametric types can be used to define a class's methods. *Parametric polymorphism* allows the construction of several functions of the same name with type parameters which distinguish them. For example, a delay function can be formulated as COST[D] with D as a parameter of different types of distance measure. In transportation research, distance can be measured as either travel time or actual distance. So the parameter D could be either a type of integer (in terms of minutes of travel time) or a type of real number (in terms of meters of physical distance). Different procedures may implement the same functionality for different argument types. In OO languages with a separate *compile* phase, this type of binding occurs at compile-time while overloading polymorphism occurs at run-time. Thus, one of the strongest advantages of parametric polymorphism is code sharing for generic types with the power of strong typing.

Encapsulation

Good OO designs enforce encapsulation and information hiding. The state of a completely encapsulated object can be manipulated and read only by invoking operations that are specified within the type definition. For example, within an object-oriented GIS, we can implement a quadtree spatial partitioning structure (Kwan et. al, 1996) but this structure can be easily replaced with an R-tree (Guttman, 1984) or a combination of a splay tree and a quadtree structure (Cobb, Chung, Shaw, and Arctur 1995) based on the data needs for faster access. Replacements, such as these, are made easier by encapsulation. Encapsulation leads to a design which is modularized at the level of the

object classes. Implementation changes will not ripple outside a class's member functions as long as all access to an object's state is by means of its public interface. In some languages (e.g., C++) the class definition may also include *private* methods which are used internally by the class but which are not accessible from outside the class.

Types of OO

When we talk about object-orientation, we accept that the concept may involve different dimensions. OO includes modeling, programming, and database management.
One widely held view is that the object-oriented approach to database management allows a more natural integration of the database with the object-oriented programming language used to manipulate it (Loomis, 1993a, Loomis, 1993b, Loomis, 1995).

a. OO modeling

OO models the real world as closely as possible (in the chosen code). It is a conceptual tool that can be used in virtually any problem domain. OO modeling refers to the development stage in which a complex system is described by a set of abstract types on which certain behaviors are defined. Unlike other modeling techniques which focus on functions, OO focuses on the essential constructs within a problem domain. The OO model views the world as objects, and starts with objects (e.g., lanes, intersections) with distinct attributes (e.g., directionality, signalization) and behaviors (e.g., linking segments or lanes and intersections to make an O-O trip). The method/message metaphor is similar enough to natural language and human problem solving that the mismatch between the conceptualization and specification is minimized.

Traditionally, transportation networks have been modeled as a planar graph (Guting, 1991). The disadvantages of a planar network include: (a) not being able to handle multiple representation levels; and (b) not handling non-planar junctions. An additional problem not specific to planar networks is that many models do not allow the network to

be a mixture of one and two way streets. It is argued that object-orientation can overcome some of these limitations. Mainguenaud (1995) modeled the logical graph topology using an object-oriented approach to allow the handling of multi-level networks. The existing digital network databases do not distinguish a local versus a regional transportation network. Mainguenaud introduces a class hierarchy where two classes are derived from the basic Node/Edge classes, called the MasterNode and MasterEdge classes. These classes override the basic functionality of the Node/Edge classes and allow the Master versions to be abstractions of sub-networks. This approach may be contrasted with other conceptualizations of *multi-scale* such as those dealing with map display (Cobb, Chung, Shaw and Arctur, 1994) In another paper, we have experimented with an object-oriented system to overcome the problems of a planar network and a network with mixed one-way and two-way directions (see Kwan, Speigle and Golledge, 1996).

b. OO programming

An OO programming language provides direct support for the OO modeling constructs. As such, the gap between system requirements and system implementation is reduced. Several different aspects of OO increase the ease of revising existing code (Carroll and Ellis, 1995). The primary mechanism is inheritance. Another mechanism is aggregation combined with delegating messages to an object's components. A third mechanism supported by some languages (e.g., C++) is that of parameterized types. A parameterized type or template class is a class which accepts as an argument in the type definition the name of another class. The template class is used to represent an algorithm (e.g., a List or Array storage class) which manipulates objects of an arbitrary type.

OO is also hailed for its ability to provide the abstractions which allow the programmer to conceptualize more difficult problems than would otherwise be possible. The levels of abstraction and encapsulation allow the programmer to design a system in terms of a set

of interacting types rather than being mired in the specifics of all of the specializations of the different types.

c. OO Database Management Systems (OODBMS)

Selection of an affordable Database Management System (DBMS) is very important in handling transportation data, especially in view of the increasing size of digital databases. An object-oriented DBMS (OODBMS) supports the object-oriented paradigm and stores the instances of classes. The systems support efficient storage of arbitrarily complex objects whose attributes are formed by inheritance and aggregation. A standard query language, however, is a topic of current research.

The limitations of existing DBMS, especially relational DBMS for storing geographic data, have been discussed by Wiegand and Adams (1994), and Kemper and Moerkotte, (1994). Wiegand and Adams (1994) began with the argument that relational models do not easily handle the complexity of geographic data and a better GIS can be built by (1) using a DBMS as a base for the system, (2) using object-oriented modeling, and (3) having an extensible system. They used an actual OODBMS and transposed current relational GIS into an OO model.

A survey of commercially available OODBMS was recently given by Kemper and Moerkotte (1994). They include Gemstone, O2, Ontos, Itasca, Versant, Matisse, Objectivity/DB, and ObjectStore. The data model, control concepts, architecture, related literature, and performance enhancements are discussed. Weigand and Adams (1994) examined several commercial OODBMS and extended-relational systems such as POSTGRES, Starburst, O2, Objectstore, and Gemstone. Advantages of OO according to their research include multiple representation (e.g., different scales), multiple geometries, and an increased amount of non-spatial information that can be stored within the feature object. Relationships between feature objects can be directly stored as part of the model

rather than via an indexing key. These points will be expanded upon in a later section in which OODBMS will be compared with relational database management systems (RDBMS) implementations.

Towards an Object-oriented GIS

Worboys (1992a & b, 1994a & b), and Worboys, Hearnshaw, & Maguire (1990), surveyed the state of the object-oriented paradigm as it applies to the handling of geo-referenced information. They outlined the major concepts behind the approach and its application in handling spatial information. These concepts have also been presented in the majority of the pure research papers, some of which will be discussed below.

Worboys (1994b) defined a geo-object, which conceptually unifies spatial, temporal, graphical and textual/numerical objects. He also pointed out that there is little use of proprietary object-oriented systems, except in cases like Milne, Milton & Smith (1993) and David, Raynal, Schorter & Mansart (1993). Research on the extended relational system has also faced difficulties. Projects in extended relational DBMS on applications of geo-referenced information include work by Lohman, Lindsay, Pirahech, & Schiefer (1991), and Rowe & Stonebraker (1987).

Some object-oriented systems have been implemented using object-oriented modeling and commercial OODBMS. Williamson & Stucky (1991) developed a generic GIS supporting Earth resource imaging analysis. The system consists of (1) a graphic, raster and text interface; (2) a database containing maps, images, and graphical and textual descriptors; and (3) a collection of processes which transform the representation and content of the data objects in the GIS. The system purportedly improved quality of reports, database updates, and analysis and allowed more timely access to widely dispersed information. They argue that a more intuitive interface with the information in the database also reduces the necessary minimum training level for analysis.

Milne et al. (1993) discuss the construction of an OO GIS based on the general purpose OODBMS, ONTOS, and use the Spatial Data Transfer Standard (SDTS) to provide a model for the basic spatial object classes. They compare the performance of ORACLE and SIRO-DBMS with their extended ONTOS system and report an impressive performance gain. David et al. (1993) describe the implementation of an OO GIS using the OODBMS O2. They distinguish between mode (semantic geographical model and localization model) and structure (spaghetti, network and maps structure). Scholl and Voisard (1992) implemented a spatial, relational-like query language on top of the OODBMS O2.

Acceptable performance from an object-oriented spatial database can be achieved only through the use of spatial indexing and clustering. These issues are receiving an increasing amount of attention in the object-oriented database literature (see Salzburg, 1994). Spatial indexing provides a way of locating objects based on spatial criteria. The numerous schemes which have been proposed include point and region quadtrees (Samet, 1989), R-trees (Guttman, 1984), and K-d trees (Bentley,1975). These schemes are essential for avoiding a serial search of the entire database when handling queries which include spatial conditionals. Clustering refers to the actual layout of a set of objects either on the permanent store or within the server-side/client-side object managers. A general discussion of the issues in clustering an object-oriented database, as well as the implementation in O2, may be found in Benzaken and Delobel(1990). The basic goal of a clustering algorithm is to place together entities which are accessed together. Implementing these schemes within an object-oriented database is complicated because both relations between objects must be taken into account as well as the class hierarchy. Full implementations would integrate the indexing completely with the query processor. An efficient indexing implementation would also be mirrored in the clustering scheme.

Many systems rely on an object-identifier scheme for retrieving objects from the database. Unless the object identifiers (OID) also indicate in some way the spatial

position of the object, this retrieval could only be implemented using non-spatial indexing. One approach is to maintain a structure which maintains a current 'working' section of the spatial database.(Cobb et al, 1995) Searches based on OID would then proceed from the most-to-least currently used spatial nodes. Another possibility is to incorporate spatial referencing directly into the OIDs.(Mainguenaud, 1995)This approach, however, would become inefficient if the spatial indices were reorganized. The capability to allow dynamic restructuring is in fact one of the more complicated aspects of the problem. Because the different commercially available OODBMS allow a developer to customize the system's indexing scheme to different degrees and because the different systems incorporate different levels of support for spatial data, these capabilities should be an important component of any assessment for purposes of procurement.

Comparison of object-oriented and relational approaches

General

Typical approaches to modeling transportation processes have used relational databases for data modeling and as the base for programming interfaces. The object-oriented approach, however, has been said to have superior modeling power because of its ability to handle complex objects, behavioral data, meta knowledge and long-duration transactions (Korth and Silberschatz, 1991). New applications require the handling of a complex object that contains other objects within it. For distinct objects, they may also need to respond in different ways to the same command and thus require the handling of behavioral data. We may also want to handle meta knowledge since often the most important data about an application are the general rules about the application rather than the specific cases. Lastly, human interactions with the data become more and more important in CAD and CASE applications. Because of these interactions, there may be

more "what if" modifications in the applications. More long-duration transactions are thus necessary. Because of these requirements there is a tendency for the use of OODBMS to expand in comparison to relational models. While the latter models simply describe system states, object-oriented models can describe both system states for data and system processes or behaviors in an integrated context (Booch, 1991; Rumbaugh, et al. 1991).

Some limitations of existing DBMS (relational) have been identified by various writers such as Herring (1992) and Wiegand & Adams (1994). First, relational RDBMS lack extensibility to provide for special application needs (e.g., provisions for the user to add new data types and methods, addition of user-defined code, design of new storage methods, and access to user-defined code). A table is created for each entity type, a row corresponds to an entity and columns contain the attribute values. However, a relational schema ends up with many additional tables because an attribute is restricted to being a simple built-in type. The two most frequently cited criticisms are that an attribute may not be a set of values and that relationships are also modeled using tables. For a GIS's spatial and non-spatial data, the data are complex enough that spreading entities and attributes into numerous tables is undesirable (Wiegand & Adams, 1994). OO modeling allows related data to be kept together and for relationships to be directly modeled (Medeiros & Pires, 1994).

The new geographic data models based on a set of feature objects provide the needed framework for a scaleless and seamless database (Mainguenaud, 1995; Guptill, 1989). It also has the DBMS characteristic of extensibility needed by GIS (Haas & Cody, 1991). For example, the ability to add new data types (e.g., points) and operations on them (e.g., distance functions) and the ability to have a new set of operations as part of the query language (e.g., overlay) are added benefits (Gunther & Lamberts, 1994). With the OO paradigm's modularity comes the capability to easily interchange access methods such as grid-files (Nievergelt, Hinterberger, & Sevcik, 1984) or R-trees (Guttman, 1984) and to

support new data storage methods so that (e.g., a map) can be stored as a quadtree. Extensibility of the optimizer allows optimizing with different data types and operators. The ability to access user-defined code and standard packages is also very important.

The use of relational databases has been popular in the past and because of this, a set of reliable and working software tools have been developed. This software includes the core database engine, modeling tools, and application development environments. An RDBMS usually incorporates a powerful query language (SQL) which has a sound mathematical basis in relational calculus (Date, 1985), and usually operates on collections of fixed format tables. As opposed to this, the tools in an object-oriented approach draw on a semantically richer background. Unfortunately this has allowed greater personalization of the approach and a standard query language has not yet emerged (c.f., Catell, 1994). As a result, two different types of object-databases have developed, one of them following many of the ideas of the relational model and the other designed to be integrated with an OO programming language (such as VERSANT). The goal of VERSANT's type of OODBMS is to transparently provide persistence to the classes defined in the OO programming language. Ties to the relational model provide enhanced relational databases with an object interface. Those tied to VERSANT or other similar languages emphasize transparent persistence of objects. Other, *middle-ware* tools operate as translator between objects on the program side, and relational sets on the storage side. Regardless of the tools selected, there is a need for more application oriented technology. The development of this software, however, must focus on questions of modularity, performance, scaleability, openness, robustness, and ease of use (Göllü, 1995).

There appear to be a limited number of formal modeling methods relevant for object-orientation. These include finite state machines (FSM) (Özveren, 1989; Ramadge & Wonham, 1987), Petri-nets and data flow diagrams (Yourdan, E., 1989; Rumbaugh, et al. 1991; and Schlaer and Melor, 1988), the Calculation of Communication Sequences (CCS) (Milner, 1980), Communication Sequential Processes (CSP) (Hoare, 1985),

Finitely Recursive Processes (FRP) (Inan & Varaiya, 1988), and Discrete Event Dynamical Systems (DEVS) (Praehofer, Auernig, & Resinger, 1993; and Zeigler, 1984).

Göllü (1995) has reviewed some examples of these software frameworks including:

a. Ptolemy. This is an object-oriented data flow based simulation tool (Ptolemy Manual, 1995). Ptolemy produces star maps in which objects called stars are represented by inputs, outputs, and pattern maps. The input output connections propagate messages that allow object interaction. Object evolution can be driven by some combination of time, events, or data tokens. It is most useful for specification and simulation of data flow among a static configuration of objects. However, it does not appear as suitable when large numbers of object relationships are concerned. It does not interface easily with other systems.

b. COSPAN. This is a general purpose software tool that provides an automatic description syntax on a set of operations. The system combines automata in order to achieve coordination of operatives in a specified way. Its logical analysis consists of symbolic testing of a system for user defined behavior and in essence its analysis constitutes a mathematical proof of the stated system behavior (Har-El & Kurshan, 1987). COSPAN's uses include the logic of discrete-event modeling in economics, and it has also been used for institutional strategic planning and epidemiological modeling in medicine.

c. CSIM. This is again a general purpose C-based process oriented environment. It is specifically designed to simulate a discrete event system and its most frequent use relates to the behavior of data communication networks (Schwetnam, 1989).

Advantages and Disadvantages of Object-oriented Systems

The advantages and capabilities of object-oriented systems include: (a) The representation of elements in the object-based environment can be categorized into a

hierarchy of object classes. The ability to perform various functions on different classes through polymorphism can further differentiate network elements and modal choices in ITS applications. This ability is not readily available without adding additional structure in the existing relational GIS. (b) An object-oriented representation will be more congenial to the perceived travel decision environment of an individual. It will therefore provide a more intuitive and user friendly environment for the implementation of an ITS. (c) It overcomes the difficulties of the planar data structure and will allow for a finer differentiation of various geographic objects in the system. Running routing and spatial search algorithms will become less problematic. (d) It will facilitate the representation and processing of a multi-level transportation network through introducing new classes across different levels and new functions for these classes; spatial search and queries will be handled more efficiently. (e) It will facilitate the integration of a wide variety of geographic objects and relations within a comprehensive geographic database.

There are also important advantages when implementing such object-oriented systems in the real world: (a) Less costly data integration - Because of the costs of acquiring and maintaining geographical data, the cost of developing an ITS can be greatly reduced if part of the data can be shared and integrated across many other applications. An object-oriented approach can greatly facilitate this through a unified scheme of object abstraction and classification. (b) Less costly maintenance and expansion - The modularity of object-oriented systems renders them highly extensible, reusable and maintainable. (c) Higher data access efficiency and reliability of the system, and (d) common use of the same database for different aspects of ITS - e.g. ATIS, ATMS, APTS.

Conclusions

The major objectives when building an object-oriented system include the establishment of an object-oriented data model through constructing the key abstractions, class structure and functions that are required. Thus, we suggest that it is possible to construct an object-

oriented GIS for ITS applications by generating a set of high-level abstractions of the ITS environment through domain analysis. These abstractions may include the transportation network, the mobile and non-mobile users of the systems, and the activity schedules and the adjustment strategies of travelers. Key mechanisms involved in the system would include processes such as data acquisition, message parsing, activity scheduling, routing, spatial search and information display (Kwan, 1994). The class structure and the module architecture of the system will then be constructed based upon the above analysis (Kwan et al., 1996). Specifically, elements of the complex transportation network (including routes that utilize various modes such as transit routes) at various spatial scales could be represented in terms of abstract data types supporting inheritance, polymorphism and dynamic binding.

Göllü (1995) argues that the validity of an object-oriented system model is likely to depend on our ability to validate the internal logic of the models, and the deployability of model component specifications. Further, he goes on to argue that the principal requirements for valid and usable software systems include:

- An ability to associate physical and logical representations (modularity)
- The ability to add new components to the system with minimal code rewrite (openness, modularity, robustness)
- The ability to collect arbitrary statistics during simulation (openness, modularity)
- The ability to run simulations with acceptable performance (performance)
- The ability to adjust simulation granularity (modularity, openness)
- The ability to simulate up to 100,000 vehicles (performance)
- The ability to specify system behavior in a straightforward language (ease of use)

(Göllü, 1995, pp. 31,32.)

Although selection of a data model and database are likely to be context dependent (i.e., different potential users will have different sets of accumulated personnel skills,

software, and data records), there appear to be increasingly strong arguments in favor of object orientation as the most versatile system for transportation modeling and planning. OO approaches provide an opportunity for the modeler to use relevant perceptual and cognitive concepts (e.g., "streets" instead of "segments"), as well as to utilize the power of concepts such as modularity, inheritance and polymorphism. In addition, the ability to handle changes as "add-ons" with object ID maintenance rather than requiring system reregistration is a positive feature. Since object orientation is an emerging approach, we have tried to emphasize both its major underlying concepts and its probable strengths. As research and development proceeds, the development of a testbed network for evaluating the claims currently being made becomes of paramount importance.

References

Bentley, J. L. (1975) Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18, 509-517.

Benzaken, V. and Delobel, C. (1990) Enhancing performance in a persistent object store: clustering strategies in O2. In A. Dearle, G. M. Shaw and S. B. Zdonik (Eds.) *Implementing Persistent Object Bases: Principles and Practices.* The Fourth International Workshop on Persistent Object Systems., pp. 403-412.

Booch, G. (1991) *Object-Oriented Design with Applications*. Redwood City, CA: Benjamin/Cummins,.

Booch, G. (1994) *Object-Oriented Analysis and Design with Applications*. Redwood City, California: Benjamin/Cummins,.

Catell, R. G. G. (Ed.) (1994*) The Object Database Standard, ODMG-93*. Morgan Kaufmann Publishers, San Francisco, CA.

Clementini, E., and Difelice, P. (1994) Object-oriented modeling of geographic data. *Journal of the American Society for Information Science,* 45, 9: 694-704.

Cobb, M., Chung, M., Shaw, K., and Arctur, D. (1995) A self-adjusting indexing structure for spatial data. *GIS/LIS Proceedings*.

Date, C.J. (1985) *An Introduction to Database Systems.* Reading, MA: Adddison Wesley.

David, R., Raynal, I., Schorter, G., and Mansart, V. (1993*) GeO2: Why Objects in a Geographical DBMS.* The Third Symposium Databases, 264-276.

Davison, P.A. (1986) Inter-relationships between British driver's age, visual attributes, and road accident histories. In A.G. Gaile, et al (Eds.), *Vision in Vehicles.* North Holland: Elsevier Science Publications, pp. 23-32.

Dingus, T.A., et al. (1989) Intentional demand requirement of an automobile moving-map navigation system. *Transportation Research A,* 23A, 4: 301-315.

Egenhofer, M.J., and Frank, A.U. (Eds.) (1989) *Object-oriented Modeling in GIS: Inheritance and Propagation.* Falls Church: American Congress on Surveying and Mapping, American Society of Photogrammetry and Remote Sensing.

Ellis, M.D., and Ellis, M.A. (1995) Designing and coding reusable C++. Reading, MA: Addison-Wesley.

Frank, A.U., and Egenhofer, M.J. (1992) Computer cartography for GIS - an object-oriented view on the display transformation. *Computers & Geosciences,* 18, 8: 975-987.

Gahegan, M.N., and Roberts, S.A. (1988) An intelligent, object-oriented geographical information system. *International Journal of Geographical Information Systems,* 2, 101-110.

Göllü, A.O. (1995) *Object Management Systems.* California PATH Research Report UCB-ITS-PRR-95-19.

Gunther, O., and Lamberts, J. (1994) Object-oriented techniques for the management of geographic and environmental data. *Computer Journal,* 37, 1: 16-25.

Gupta, R. and Horowitz, E. eds. (1991) *Object-oriented Databases with Applications to CASE, Networks, and VLSI CAD.* New Jersey: Prentice Hall.

Guptill, S. C. (1989) *Speculations on Seamless, Scaleless Cartographic Data Bases.* Auto-Carto 9, Ninth International Symposium on Computer-Assisted Cartography, Baltimore, Maryland, 436-443.

Guting, R. H. (1991) *Extending a Spatial Database System by Graphs and Object Class Hierarchies.* Geographic Database Management Systems Workshop Proceedings, Capri, Italy.

Guttman, A. (1984) *R-Trees: A Dynamic Index Structure for Spatial Searching.* ACM International Conference on Management of Data.

Haas, L. M., and Cody, W. (1991). Exploiting Extensible DBMS in Integrated Geographic Information Systems. Advances in Spatial Databases. In O. Gunther and H. Schek (Eds.) *Report from Second Symposium*, SSD91. New York: Springer-Verlag, 423-450.

Har-El, Z., and Kurshan, R.P. (1987) *COSPAN Users guide.* Murray Hill, NJ: AT&T Bell Laboratories.

Herring, J.R. (1992) Tigris - a data model for an object-oriented geographic information system. *Computers & Geosciences,* 18, 4: 443-452.

Hoare, C.A.R., (1985) *Communicating Sequential Processes.* Englewood Cliffs, NJ: Prentice-Hall International.

Inan, K., and Varaiya, P. (1988) *Finitely recursive process models for discrete event systems.* IEEE Transactions, Auto.Control, AC-33, 7: 626-639.

Jackson, R. W. (1994). Object-oriented modeling in regional science: an advocacy view. *Papers in Regional Science*, 73,4: 347-367.

Jonah, B.A., and Dawson, N.E. (1987) Youth and risk: Age differences in risky driving, risk perception and risk utility. *Alcohol, Drugs, and Driving,* 3, 3-4: 13-29.

Kanninen, B. J. (1996) Intelligent transportation system: an economic and environmental policy assessment. *Transportation Research-A*, 30(1), 1-10.

Kemp, K. (1992) *Environmental Modelling with GIS: A Strategy for Dealing with Spatial Continuity.* GIS/LIS Annual Conference, 397-406.

Kemper, A. and Moerkotte, G. (1994) *Object-oriented Database Management Applications in Engineering and Computer Science*. New Jersey: Prentice Hall.

Khoshafian, S. (1993). *Object-oriented Databases* New York: Wiley.

Khoshafian, S., and Abnous, R. (1990*). Object-orientation: Concepts, Languages, Databases, User Interfaces*. New York: Wiley..

Kim, W. (1990) *Introduction to Object-oriented Databases*. Cambridge, MA: MIT Press.

Korth, H. F. and Silberschatz, A. (1991) *Database System Concepts*. Singapore, McGraw-Hill, Inc.

Kwan, M.-P. (1994). *GISICAS: A GIS-Interfaced Computational-Process Model for Activity Scheduling in Advanced Traveler Information Systems*. University of California, Santa Barbara.

Kwan, M.-P., Speigle, J., and Golledge, R.G. (1996*) Developing an Object-Oriented testbed for modeling transportation systems*. Discussion paper, Research Unit in Spatial Cognition and Choice (RUSCC), Department of Geography, University of California Santa Barbara.

Lohman, G., Lindsay, B., Pirahesh, H., and Schiefer, K. B. (1991). Extensions to Starburst: Objects, Types, Functions and Rules. *Communications of the Association for Computing Machinery*, 34, 94-109.

Loomis, M.E.S. (1993a) Object programming and database management. *Journal of Object-Oriented Programming*, February, 75-79

Loomis, M.E.S. (1993b) Object programming and datebase management: Differences in perspective between the two. *Journal of Object-Oriented Programming*, May, 31-34, 67.

Loomis, M.E.S. (1995) *Object databases, the essentials*. New York: Addision-Wesley,.

Mainguenaud, M. (1995) Modelling the network component of geographical information systems. *International Journal of Geographic Information Systems*, 9, 575-593.

Martin, J. (1993) *Principles of Object-oriented Analysis and Design*. New Jersey: Prentice Hall.

Medeiros, C., and Pires, F. (1994). Databases for GIS. *SIGMOD Record*, 23(1), 107-115.

Milne, P., Milton, S., and Smith, J.L. (1993) Geographical object-oriented databases - a case study. *International Journal of Geographical Information Systems,* 7, 1: 39-55.

Milner, R. (1980) *A calculus of communicating systems.* New York: Springer-Verlag.

Nievergelt, J., Hinterberger, H., and Sevcik, K. (1984) *The Grid-File: An Adaptable Symmetric Multikey File Structure.* ACM Trans. on Database Systems.

Noy, I.Y. (1990) *Attention and performance while driving with the auxiliary in-vehicle displays.* Road Safety and Motor Vehicle Regulation - Transport Canada. Ottawa: Report #TP10727(E), December.

Özveren, C.M. (1989) *Analysis and control of discrete event dynamic systems: A state space approach.* Ph.D. dissertation, MIT.

Praehofer, H., Auernig, F., and Resinger, G. (1993) An environment for DEVS-based multiformalism simulation in common LISL/CLOS. *Discrete Event Dynamic Systems: Theory and Application,* 3, 2: 119-149.

Ptolemy Manual (1995) *The almalmagest,* Volume 1-4, Version 0.5.2, College of Engineering, UC Berkeley.

Ramadge, P., and Wonham, W. (1987) Supervisor control of a class of discrete event processes. *SIAM Journal of Control Optimization,* 25, 1: 206-230.

Roberts, S.A., and Gahegan, M.N. (1993) An object-oriented geographic information system shell. *Information and Software Technology,* 35, 10: 561-572.

Roberts, S.A., Gahegan, M.N., Hogg, J., and Hoyle, B. (1991) Application of object-oriented databases to geographic information systems. *Information and Software Technology,* 33, 1: 38-46.

Rowe, L. A., and Stonebraker, M. R. (1987) *The Postgres Data Model.* The 13th VLDB Conference, San Mateo, California, 83-96.

Rumbaugh, J., Blaha, M., Premerlani, W.E.,and Lorensen, W. (1991) *Object-oriented modeling and design.* Englewood Cliffs, NJ: Prentice-Hall.

Salzberg, B. (1994). On indexing spatial and temporal data. Information *Systems*, 19, 447-465

Schlaer, S., and Mellor, S. (1988) *Object-oriented systems analysis: Modeling the world in data.* Englewood Cliffs, NJ: Yourdan Press.

Schofer, J. L., Khattak, A. and Koppelman, F. S. (1993) Behavioral Issues in the Design and Evaluation of Advanced Traveler Information Systems. *Transportation Research-C,* 1, 2: 107-117.

Scholl, M., and Viosard, A. (1992) *Object-oriented database systems for geographic applications: An experiment with O2.* Paper presented at the Geographic Database Management Systems Workshop Proceedings, Capri, Italy. May 1991.

Schwetman, H. (1989) *CSIM Reference Manual* (Revision 13). Micro Electronics and Computer Technology Corporation, 3500 West Valcones Center Drive, Austin, Texas 78759.

Shinar, D. (1978) *Psychology on the road - the human factor in traffic safety.* New York: John Wiley & Sons.

Tomlin, C. D. (1990). *Geographic Information Systems and Cartographic Modelling.* Englewood Cliffs, New Jersey: Prentice-Hall,.

van Oosterom, P., and van den Bos, J. (1989) An object-oriented approach to the design of geographic information systems. *Computers & Graphics,* 13, 4: 409-418.

Walker, J., Alicandri, E., Sedney, C., and Roberts, K. (1990) *In-vehicle navigation devices: Effects on the safety of driver performance.* Report #FHWA-RD-90-053. Office of Safety and Traffic Operations, Research and Development, Federal Highway Administration, McLean, VA, May.

Watling, D., Vuren, T. V. (1993) The Modeling of Dynamic Route Guidance Systems. *Transportation Research-C,* 1(2), 159-182.

Wiegand, N., and Adams, T. M. (1994). Using Object-Oriented Database Management for Feature-Based Geographic Information Systems. *Journal of the Urban and Regional Information Systems Association,* 6(1), 21-36.

Williamson, R., and Stucky, J. (1991). An Object-Oriented Geographical Information System. In R. Gupta and E. Horowitz (eds.) *Object-Oriented Databases with Applications to CASE, Networks and VLSI CAD,* Englewood Cliffs, NJ: Prentice-Hall, 297-311.

Worboys, M.F. (1992a) A generic model for planar geographical objects. *International Journal of Geographical Information Systems,* 6, 5: 353-372.

Worboys, M. F. (1992b). *Object-Oriented Models of Spatiotemporal Information.* GIS/LIS Proceedings, 2, 825-834.

Worboys, M.F. (1994a) A unified model for spatial and temporal information. *Computer Journal,* 37, 1: 26-34.

Worboys, M.F. (1994b) Object-oriented approaches to geo-referenced information. *International Journal of Geographical Information Systems,* 8, 4: 385-399.

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

Yourdan, E. (1989) *Modern structural analysis.* Englewood Cliffs, NJ: Yourdan Press.

Zeigler, B. (1984) *Multifaceted modeling and discrete event simulation.* London: Academic Press.

Figures

Figure 1: Different elements in Intelligent Transportation Systems

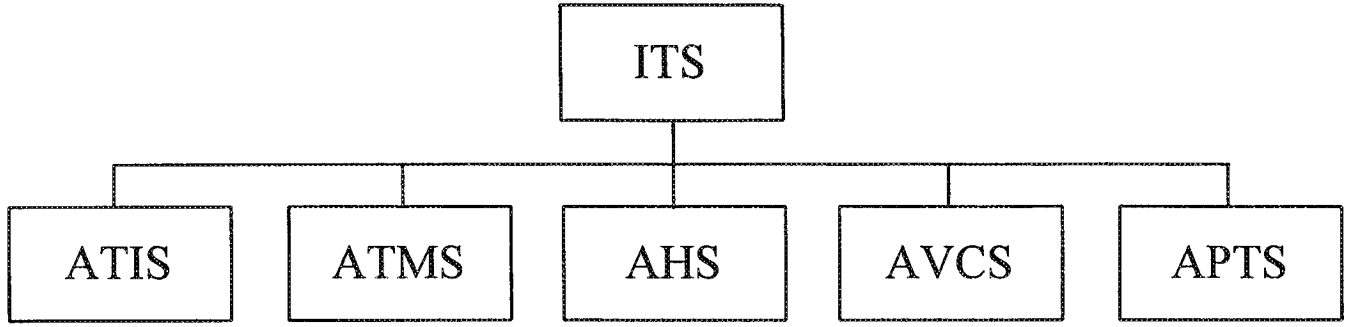


Figure 2

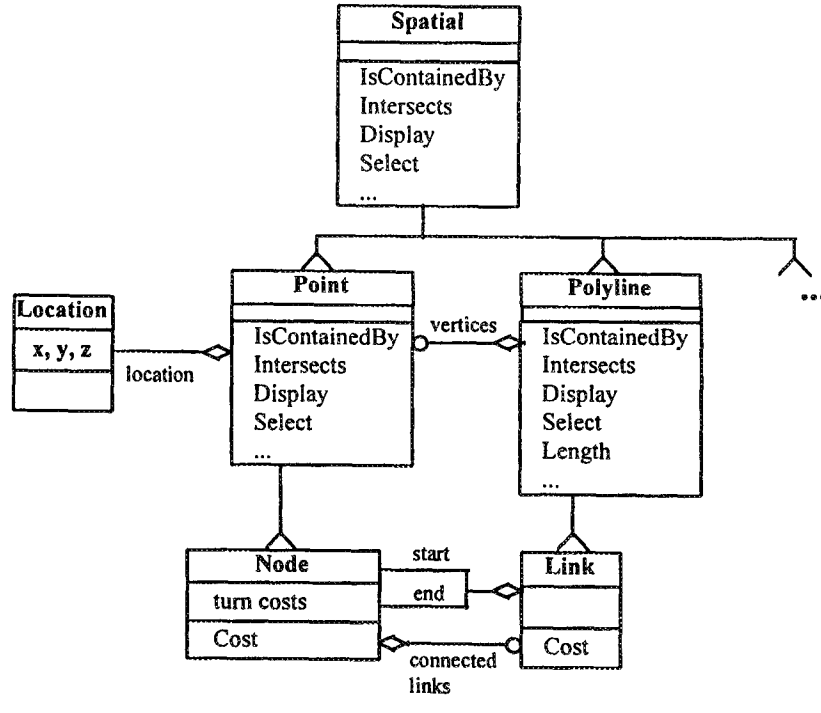

Spatial
IsContainedBy
Intersects
Display
Select
...
Location
x, y, z
location
Point
IsContainedBy
Intersects
Display
Select
...
vertices
Polyline
IsContainedBy
Intersects
Display
Select
Length
...
...
Node
turn costs
Cost
start
end
connected
links
Link
Cost

Figure 3

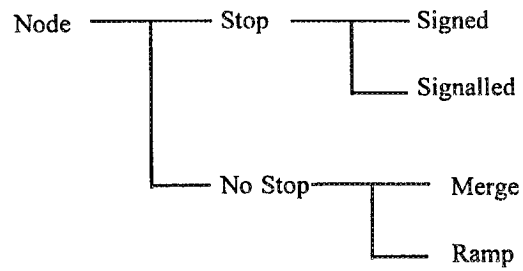


Figure 4

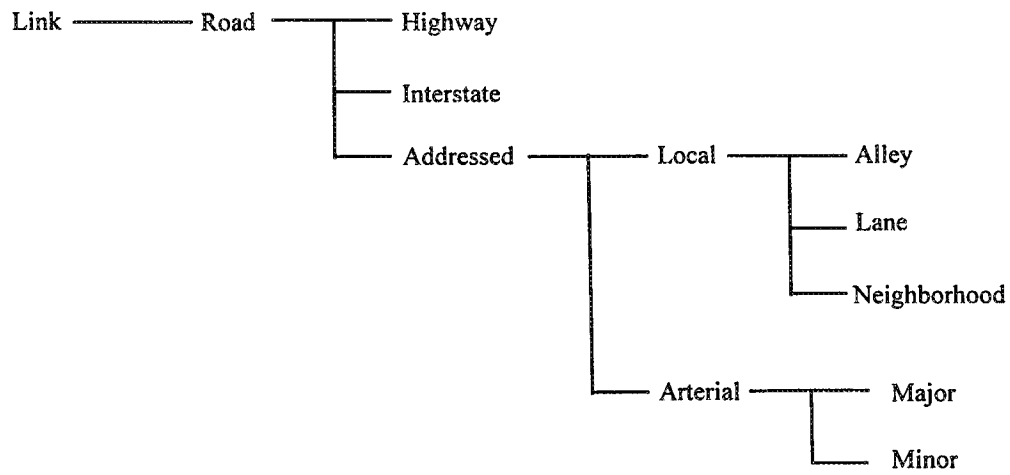
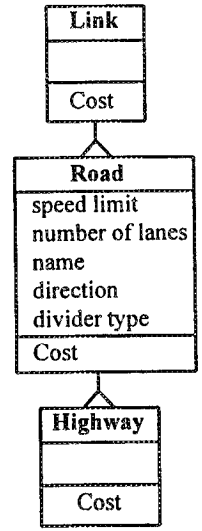

Link
Road
Highway
Interstate
Addressed
Local
Alley
Lane
Neighborhood
Arterial
Major
Minor

Figure 5


Link
Cost
Road
speed limit
number of lanes
name
direction
divider type
Cost
Highway
Cost