Ontologies and Knowledge Sharing in Urban GIS

Frederico T. Fonseca[1] Max J. Egenhofer[1] Clodoveu A. Davis Jr. [23] Karla A.V. Borges [3]

[1]National Center for Geographic Information and Analysis
and
Department of Spatial Information Science and Engineering
University of Maine
Orono, ME 04469-5711, USA
[2]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte MG Brazil
[3]Prodabel - Empresa de Informática e Informação do Município de Belo Horizonte
Belo Horizonte MG Brazil

ABSTRACT

Data and knowledge exchange among users of urban information systems presents many challenges. This paper discusses issues related to the use of ontologies in the development of urban geographic information systems and proposes the creation of software components from diverse ontologies as a way to share knowledge and data. These software components are derived from ontologies using an object-oriented mapping. The translation of an ontology into an active information system component leads to *ontology-driven information systems* and, in the specific case of geographic applications, to *ontology-driven geographic information systems*. We analyze the urban environment from the ontologists' point of view and make some inferences about the relationship between knowledge and data sharing, and the theory of *bona fide* and *fiat* objects. We also discuss implementation issues, such as the use of Ontolingua as an ontology editor, and CORBA IDL generator, CORBA, and Java as object platforms.

1 INTRODUCTION

Starting an urban geographic information system (GIS) project presents many challenges. Describing the detail-rich urban environment is one of them. To face this challenge, the use of existing knowledge from previous GIS projects is a necessity. Beyond that, the use of existing data is also desirable. But the lack of formal methods to reuse knowledge and data makes this task difficult. We propose in this paper the creation of software components from diverse ontologies as a way to share knowledge and data. These software components are implemented as classes derived from ontologies, using an object-oriented mapping. The use of an ontology, translated into an active information system component, leads to *Ontology-Driven Information Systems* (ODIS) (Guarino 1998) and, in the specific case of GIS, it leads to what we call *Ontology-Driven Geographic Information Systems* (ODGIS). The system architecture for an ODGIS has been described elsewhere (Fonseca and Egenhofer 1999).

1.1 Knowledge Exchange in Urban GIS

The exchange of data and knowledge among urban GIS users has multiple aspects. While the reuse of data tends to be done horizontally, the reuse of knowledge can be done vertically and horizontally. For instance, exchanging data about a city, one must notice that urban data are strongly related to location and, therefore, it is likely that a city will need data on neighboring cities, because the urban environment does not cease to exist abruptly at the municipal borders. Issues like transportation and environmental concerns can only be

treated in a continuous way. The sudden interruption of a public transportation line because it reached the limits of two cities is an artificial expedient and does not contribute to solve the actual problem. On the other hand, urban knowledge can operate nicely within these kinds of borders. Within a defined group of users, such as cities in a state, a well-defined conceptualization of an urban environment can be shared among all the cities of the state even if they are not neighbors. The concept of *geospatial information communities* (GIC) as a group of users that share a digital geographic information language and spatial feature definitions was introduced by McKee and Buehler (1996). Bishr *et al.* (1999) revised this concept considering a GIC to be a group of spatial data users and producers who share an ontology of the geographic world. This agrees with Guarino (1998) who distinguishes an application ontology from a generic knowledge base by considering the latter as a particular knowledge base that describes facts always true for a community of users. The revised GIC concept is fundamental to ODGISs.
An urban ontology therefore comprises:

- objects (e.g., legal and administrative regions, streets, blocks, parcels, schools);
- relations (e.g., a school manages a school district, a parcel belongs to a block);
- events (e.g., traffic accidents, infrastructure maintenance); and
- processes (e.g., noise pollution, traffic flow).

This paper does not intend to specify ontologies. Our intention here is to show how urban ontologies can be used to generate software components. These components enable different levels of data sharing and knowledge reuse.

1.2 Ontology

Gruber (1992) defines an ontology as an explicit specification of a conceptualization. Guarino (1998), while agreeing with Gruber, presents a refined distinction between an ontology and a conceptualization: an ontology is a logical theory accounting for the *intended meaning* of a formal vocabulary, i.e., its *ontological commitment* to a particular *conceptualization* of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. This commitment and the underlying conceptualization are reflected in the ontology by the approximation of these intended models. Guarino (1998) advises against using ontology just as a fancy name denoting the result of activities like conceptual analysis and domain modeling.

Although the use of ontologies started with Artificial Intelligence, ongoing research on ontology can be found throughout the computer science community, in such areas as computational linguistics and database theory. It covers fields ranging from knowledge engineering, information integration, and objected-oriented analysis to such applications as medicine, mechanical engineering, and GIS. Frank (1997) discusses the use of explicit ontologies in systems development. Using ontologies to build GIS applications can help data integration and avoid problems, such as inconsistency between *ad-hoc* ontologies built into the system. However, there is a gap between the ontologies and the software components. To allow transfer of knowledge from ontologists (i.e, the specialists in the area of the application) to software engineers it is necessary to focus on the consistent part of an ontology instead of highlighting differences between ontologies. It is also necessary to exclude the historical and philosophical point of view. Both the engineering and the cognitive views of the world are necessary to produce the small theories that account for the behavior of certain parts of reality. The first is necessary to integrate engineering knowledge into the system, the second to make it understandable to the user at the interface (Frank 1997). Ontology plays an essential role in the construction of GIS, since it allows the establishment of correspondences and interrelations among the different domains of spatial entities and relations (Smith and Mark 1998).

1.3 An Object View of the World

The use of the object data model as the basic conceptualization of space has been discussed in (Nunes 1991). Nunes (1991) considers that the issue of defining geographic space is actually the issue of defining and studying the geographical objects, their attributes and relationships. The object view of the geographic world is established in (Egenhofer and Frank 1992). An outline of the main concepts behind the object-oriented approach and its application to geo-referenced information handling appears in (Worboys 1994). Here we introduce basic concepts of object orientation.

A *class* is the extension of the concept of an abstract type, a structure that represents a single entity, describing both its information contents and its behavior. Every object is an instance of a class, which defines a structure and a set of operations that are common to a group of objects (Chin and Chanson 1991). An object functions as a complex data structure, that is capable of storing all of its data, along with information about the necessary procedures to create, destroy and manipulate itself. In an object-oriented GIS, each object instance contains its graphical characteristics, its geographic location, and all of the associated data.

The ability to hide from the user the internal structure of an object is called *encapsulation*. With encapsulation, it is only possible to manipulate the object's data using a set of predefined functions, thus ensuring data independence. The internal definition of the data structure can then change, without influencing what the user perceives

Classes are often defined hierarchically, taking advantage of one of the most important concept in object-oriented systems: *inheritance*. It is possible to define more general classes, containing the structure of a generic type of object, and then specialize this class by creating subclasses. The subclasses will inherit all properties of the parent class and add some more of its own. For instance, within a local government you can have different views and uses for land parcels. A standardization committee can specify a land parcel definition with general characteristics. Each department that has a different view of a land parcel can specify its own land parcel class, inheriting the main characteristics from the general definition of land parcel and including the specifics of the department. When one class inherits directly from only one class, it is called *single inheritance*, and when a class inherits from more than one class, it is called *multiple inheritance* (Cardelli 1984). Multiple inheritance is a controversial concept, with benefits and drawbacks (Tempero and Biddle 1998). It is hard to combine inherited traits from several ancestors. A variation of the inheritance scheme is *delegation*, a mechanism in which an object delegates the responsibility for servicing an invocation request to another object (Chin and Chanson 1991). Although the implementation and use of multiple inheritance is non-trivial (Tempero and Biddle 1998), its use in geographic data modeling is essential (Egenhofer and Frank 1992). ODGIS makes an extensive use of object-oriented techniques, specially multiple inheritance. The combined use of objects and ontologies provides a rich model to represent geographic entities, avoiding the problems of poor representation pointed out in (Nunes 1991).

This work also extends the concept of interoperability. Interoperability is defined as the ability of a system or its components to share information and applications (Bishr 1997). We propose here to extend this concept to include also knowledge sharing throughout the life cycle of an information system, including development and use.

The remainder of this paper is organized as follows. Section 2 gives an overview of related work on the use of ontology to support information sharing. Section 3 analyzes the urban environment from the ontologists' point of view. Section 4 describes the mapping of classes from ontologies. Section 5 discusses some implementation issues. Section 6 presents conclusions and future work.

2 RELATED WORK

The use of ontology in information system building is thoroughly discussed by Guarino (1998), and specifically in GIS building by Frank (1997) and Smith and Mark (1998). Gruber (1991) suggested that ontologies play a software specification role. Also Nunes (1991) pointed out that the first step to build a next-generation GIS would be the construction of a systematic collection and specification of geographic entities, their properties, and relations. However, philosophers and software engineers have different perspectives about ontologies. In this section we review work that deals with the use of ontology in the development of information systems, with knowledge sharing, and with ontology-related system architectures.

2.1 Ontology and Software Development

According to Guarino (1998), there is a difference in the definition of ontology in the philosophical sense and in the way the term is used in the Artificial Intelligence (AI) field. In AI, ontology is seen as an engineering artifact that describes a certain reality with a specific vocabulary, using a set of assumptions regarding the intended meaning of the vocabulary words. Meanwhile, in the philosophical arena, ontology is characterized as a particular system of categories reflecting a specific view of the world. Smith (1998) notes that since, to the philosopher, ontology is the science of being, it is inappropriate to talk about a plurality of ontologies, as engineers do. To solve this problem, Smith (1998) suggests a terminological distinction between referent-based or reality-based ontology (R-ontology) and elicited or epistemological ontology (E-ontology). R-ontology is a theory about how the whole universe is organized and corresponds to the philosopher's point of view. An E-ontology fits the purposes of software engineers and information scientists and can be defined as a theory about how a given individual (or group or language or science) conceptualizes a given domain. There are as many proper E-ontologies as there are GICs.

In order to build software components from ontologies, it is reasonable to assume that ontologies are available on the market. As ontology development technology evolves, the benefits of ontology use will outweigh the costs of developing them. With the success of this technology, large-scale repositories of ontologies will be available in diverse disciplines (Farquhar *et al.* 1996), and previous work has been developed based upon this availability assumption (Kashyap and Sheth 1996). As Frank (1997) assumes, we believe that there is a commercial production of ontologies, and that these ontologies are good enough to be used. This position is not shared by Guarino (1998), however, who believes that the available quantity of ontological knowledge is modest, although of good quality. Kemp and Vckovski (1998) consider that although certain types of geographic phenomena, like discrete objects, have been the object of ontology study, spatially continuous phenomena, like temperature and soil moisture, have received little attention. Guarino (1998) suggests that it is more feasible today to use very generic ontologies, although this solution has the drawback of limiting the degree of reusability of the software components and knowledge. The other option is to use an ontology library containing specialized ontologies of domains and tasks. The translation of this library into software components reduces the cost of conceptual analysis and ensures the ontological adequacy of the information system. The solution presented here tries to shorten the gap between these two solutions by allowing navigation through ontologies of different specialization levels.

2.2 Knowledge Interoperation

The base of ODGIS is the willingness of users to share knowledge and data. The reasons to do so can be economic or regulatory. Reusing data can decrease dramatically the costs of developing a GIS project. Reusing knowledge may decrease costs but, more importantly, it may mean the success of a project (Huxhold 1991). Neches *et al.* (1991) suggest that it is difficult to lower these costs and it is better to focus research on sharing the acquired knowledge. Sharing is the only way to build qualitatively bigger knowledge-based systems, because we can rely on previous labor and experience. Some high-level government institutions recommend the use of mechanisms that enhance the possibility of data sharing (Arctur *et al.* 1998).

For interoperability to take place, an agreement on the terminology in the shared area must occur through the definition of an ontology for each domain (Wiederhold 1994). Ontologies are crucial for knowledge interoperation and they can serve as the embodiment of a consensus reached by a professional community (Farquhar *et al.* 1996). Sharing the same ontology is a pre-condition to data sharing and data integration. Kashyap and Sheth (1996) consider that there should be an ontological commitment revealing the agreement between the generic user querying the database and the database administrator that made data available. In ODGIS, the agreement is expressed through the use of *elected ontologies* that are used to build new ontologies, from which the software components will be derived. An alternative to an explicit ontological commitment is the semantic approach. Bergamaschi et *al.* (1998) propose the derivation of a global schema to overcome the absence of a common shared ontology through the use of clustering techniques. The solution of semantic heterogeneities is done through description logic. Rodriguez *et al.* (1999) present a similarity assessment among ontologies using a feature-matching process and semantic distance calculations.

To clarify the idea of who are the users, it is better to group them in geospatial information communities (GIC). Bishr (1999) considers that the definition of a GIC should not be restricted to the data model sharing, but we should use common ontologies as the high-level language that holds those communities together. Therefore, a GIC is a group of users that share an ontology. In the solution that is presented here, we allow the GIC to commit to several ontologies. The users have means to share the common ontologies through the use of classes derived from ontologies.

Bishr (1998) distinguishes three types of heterogeneity: semantic heterogeneity, in which a fact can have more than one description; schematic heterogeneity, in which the same object in the real world is represented using different concepts in a database; and syntactic heterogeneity, in which the databases use different paradigms. Semantic heterogeneity should be solved before schematic and syntactic heterogeneity. The use of *semantic translators* is suggested as the means to provide interoperability among and within GICs. Semantic translators, also called mediators (Wiederhold 1991), use a common ontology library as a measure of semantic similarity. Dynamic approaches for information sharing, as provided by semantic translators, are more powerful than the current approaches that promote standards (Bishr 1997).

Wiederhold (1994) also proposes mediation as the principal means to resolve semantic heterogeneity through an incremental domain approach that brings domains together when needed. Mediators look for geographic data and translate data into a format understandable by the end user. The mediators are pieces of software with embedded knowledge. Experts build the mediators by putting their knowledge into them and keeping them up to date.

The solution we present here addresses schematic and syntactic heterogeneity and proposes the use of multiple inheritance from diverse ontology classes to solve this issue. Semantic heterogeneity is solved when GICs commit to the use of the same ontology.

2.3 Ontology-Based System Architectures

The need to share geographic information is well documented (McKee and Buehler 1996; Vckovski 1998; Goodchild *et al.* 1999). In the past, exchanging geographic information was simple as sending paper maps or raw data tapes through mail. Today, there is a huge amount of data gathered about the Earth, computers throughout the world are connected, and the use of GIS has become widespread. Although spatial information systems have been characterized as an integration tool, GIS interoperability is far from being fully operational (Vckovski *et al.* 1999).

Literature shows many proposals for integration of data, ranging from federated databases with schema integration (Sheth and Larson 1990) and the use of object orientation (Kent 1994; Papakonstantinou *et al.* 1995), to mediators (Wiederhold 1991) and ontologies (Wiederhold 1994; Guarino 1998). Sheth (1999) considers that the new generation of information systems should be able to solve semantic heterogeneity to make use of the amount of information available with the arrival of Internet and distributed computing. The support and use of multiple ontologies should be a basic feature of these modern systems. We review here a ontology editor that can be used for these kind of system and a information retrieval system based on ontologies.

2.3.1 Ontolingua

A mechanism to edit, browse, translate, and reuse ontologies is presented in the Ontolingua Server (Farquhar *et al.* 1996). This work is based on Ontolingua (Gruber 1992), a language to specify ontologies. The syntax and semantics of Ontolingua definitions are based on the Knowledge Interchange Format (KIF) (Genesereth and Fikes 1992). KIF is a monotonic first-order predicate calculus with a simple syntax and support for reasoning about relations. The adopted approach is to translate ontologies specified in a standard, system-independent form into specific language representations. The Ontolingua Server allows multiple users to collaborate on ontology construction in a shared section. It also accepts queries from remote applications. The Ontolingua translation strategy allows the use of an ontology both in the development and in the production phases of a system. The translation targets can be representations as CORBA interface definition language (IDL) (OMG 1991), Prolog (Clocksin and Mellish 1981), Epikit (Genesereth 1990), and KIF.

2.3.2 OBSERVER

OBSERVER (Kashyap and Sheth 1996; Mena *et al.* 1996; Mena *et al.* 1998) is an architecture for query processing in global information systems that supports interoperation across ontologies. OBSERVER focuses on information content and semantics, and employs a loosely-coupled approach to match different vocabularies used to describe similar information across domains. Instead of integrating pre-existing ontologies, OBSERVER uses synonym relationships between terms across ontologies. Synonymy, hyponymy, and hypernymy are semantic relations defined between words and word senses. Synonymy (*syn* same, *onyma* name) is a symmetric relation between word forms. Hyponymy (sub-name) and its inverse, hypernymy (super-name), are transitive relations between sets of synonyms. This semantic relation is usually organized in a hierarchical structure (Miller 1995). OBSERVER uses hyponymy and hypernymy to translate terms that are not synonymous in different ontologies. It substitutes non-translated terms with the intersection of their immediate parents or the union of the immediate children.

The basic components of OBSERVER are the query processor, the ontology server, and the interontology relationships manager. The user query is based on an ontology chosen by the user. The query processor matches the terms in the user ontology to the system component ontologies. The information about ontologies is provided by the ontology server, using mappings between ontologies and the structures in data repositories. The synonym relationships are provided by the interontology relationships manager.

3 THE URBAN ENVIRONMENT

In this section we analyze the urban environment from the ontologists point of view. We use the work of Smith (1995) and Smith and Mark (1998) to identify ontology issues for urban environments. But first we make some inferences about the relationship between knowledge and data sharing, and Smith's *bona fide* and *fiat* objects theory.

3.1 *Bona Fide* and *Fiat* Objects

Smith introduces *bona fide* and *fiat* objects, a classification based on boundary characteristics of geographic features. *Bona fide* objects have boundaries that exist independently of all human cognitive acts. These boundaries are parts of the geographic objects, like riverbanks or coastlines. *Fiat* boundaries, on the other hand, correspond to no genuine heterogeneity on the side of the geographic entities. They are abstract, created by acts of human decision, and usually related to laws or political decrees.

The urban environment is very complex and although it has some natural occurrences like rivers that are *bona fide* objects, it is essentially made of *fiat* objects. Even features such as rivers, when crossing urban environments, have their boundaries shaped by people and can be considered as *fiat* objects.

3.2 Boundaries

Smith also introduces a general typology of spatial boundaries. Boundaries can be crisp or indeterminate, complete and incomplete, enduring and transient, symmetrical and asymmetrical. Although most of the urban objects have crisp, complete, and enduring boundaries, some of them are different. Processes like noise pollution can be related to sources that possess time-related, variable characteristics, such as traffic noise during the course of a day, and thus have an indeterminate and transient boundary. Another example of indeterminate urban boundaries is the case of neighborhoods. Some regions near the border can be considered inside a certain neighborhood or inside the adjacent one, depending on which department or person assesses it. Depending on where a property is located, in the nice neighborhood A or in the adjacent but not so nice neighborhood B, there can be a difference in property value. The problem of implementing these different boundaries in a GIS can be solved by using an ontology of boundaries in an ontology-driven geographic information system.

3.3 Mereotopology

A theory of part and whole, or mereology (Smith and Mark 1998) is important because geographic objects are typically complex and have multiple constituent parts. In urban systems, where most of the objects are *fiat* objects that are derived from hierarchic structures, this importance is even clearer. Topology (Alexandroff 1961) is important because geographic objects prototypically can be connected or contiguous, scattered or separated, closed or open. Smith (1993) suggests mereotopology, that is, an alliance of topological methods with the ontological theory of part and whole. The study of the ontology of geographic kinds highlights certain characteristic types of distortions that are involved in our cognitive relations to geographic phenomena. Geographic information systems need to manipulate representations of geographic entities, and the ontological study of the corresponding entity types, especially those at the basic level, provide default characteristics for such systems. Entity types present in ontologies can be used to improve the way data are exchanged either in the semantic or the representation aspects. Furthermore, the ontology of the geographic space, of the geographic objects, and of the phenomena of the geographic space is different from other ontologies, because topology and part-whole relations play a major role in the geographic domain.

3.4 Knowledge and Data Sharing in Urban Environments

Smith (1995) observes that most of us live in a hierarchy of fiat objects. These objects are specified in ontologies related to political divisions. A national land use ontology can be defined in this fashion, initially at a high level. More detailed ontologies can be specified for other levels in the political hierarchy. At the national level, only large political divisions are recognized, along with federal legislation aspects on property ownership. At the next level, a state land use ontology would deal with specific details related to state legislation. Counties and municipalities can refine these notions further, adding specific knowledge-related local characteristics. Knowledge sharing is more likely to occur over this vertical axis, through the action of national associations like URISA, which can gather information and specify high-level ontologies. Guarino (1998) classifies these high-level ontologies as top-level, domain, or task ontologies. This kind of knowledge is easily transferable to information communities all over this vertical axis.

On the other hand, data sharing is more related to *bona fide* objects. The geographic location of these objects has a greater influence on data exchange than the political hierarchy, which determine *fiat* objects. When perceived at a large scale, even *fiat* objects, such as cities tend to behave more like *bona fide* objects with distinct physical characteristics. A county-level environmental study will consider data on neighboring towns, and a regional study will also consider geographically close states.

4 ONTOLOGY-DRIVEN GEOGRAPHIC INFORMATION SYSTEMS

In ontology-driven geographic information systems (ODGIS), an ontology is a component, such as the database, cooperating to fulfill the system's objectives. The first step to build ODGIS is to specify the ontologies using an ontology editor. The editor should be able to translate the ontologies into a formal language. The translated ontologies are available to be browsed, and therefore can be employed to provide the user with information about the knowledge embedded in the system. They can also be used as classes that contain data and operations that constitute the system's functionality.

ODGIS are built using software components derived from various ontologies. These software components are classes that can be used to develop new applications. Being ontology-derived, these classes embed knowledge extracted from ontologies. We present here the result of this mapping, that is, the classes that are generated from ontologies. Other system components, such as the user interface and the underlying database, are not discussed here.

4.1 Ontology and User Classes

The application developer can derive new classes, more specific to the application, called *user classes*, which are different from more generic *ontology classes*. The user classes belong to the application ontologies level, while ontology classes belong to top-level, domain, or task ontologies (Guarino 1998). An application developer can build an application using either ontology classes or user classes, although they are separated here for clarification purposes (Figure 1).

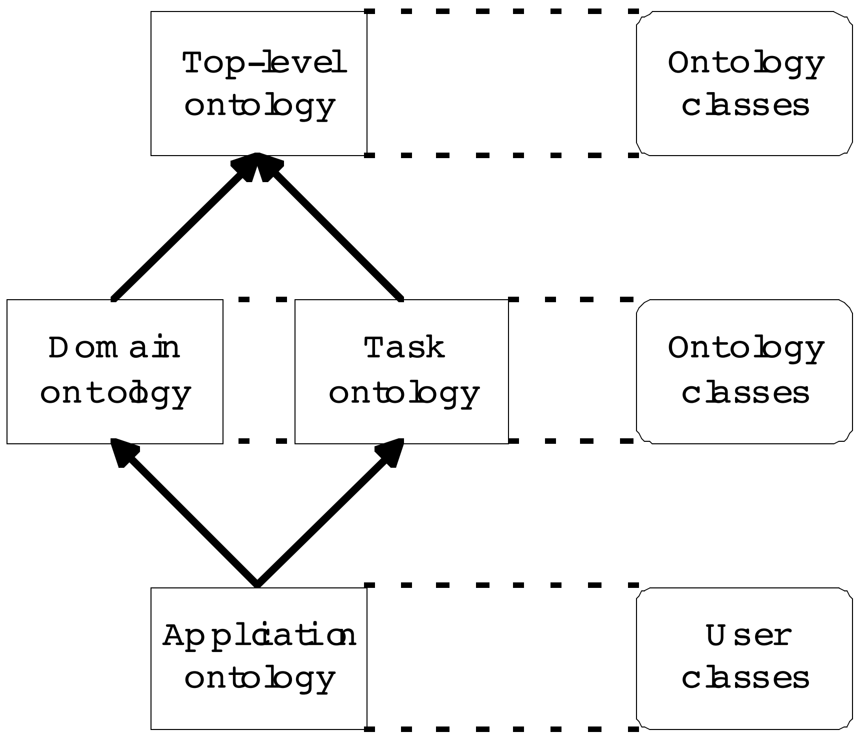

Figure 1 - Ontology and user classes, extended from Guarino (1998)

After building ontology classes it is important to have a mechanism that enables the application developer to create user classes. User classes represent objects that have diverse characteristics. For instance, land parcels have geometric characteristics along with alphanumeric attributes. Some models define these geometric characteristics within the parcel class, like in the feature model of OpenGIS (McKee and Buehler 1996). We propose here to use multiple inheritance (Cardelli 1984) to define these kinds of classes. The use of multiple inheritance allows the developer to make use of the existing ontologies to build new classes. In the example, instead of defining a `User Parcel` as a class that includes geometric data, we believe that the `User Parcel` should descend from `Parcel` and `Geometric` classes. Therefore, instead of having a user parcel that contains its own geometry definition, there would be a class that inherits geometric characteristics from a geometric class, such as polygon, and inherits application-specific characteristics from a more generic parcel class. This approach allows methods that deal with geometric objects to have direct access to geometry of the `User Parcel`, instead of relying on the `User Parcel` class to handle it over to the geometric methods. This solution is achieved through *interface conformance*. A class should conform to every parent class. So the `User Parcel` in our example can be seen and treated as both a `Polygon` and a `Parcel` (Figure 2).

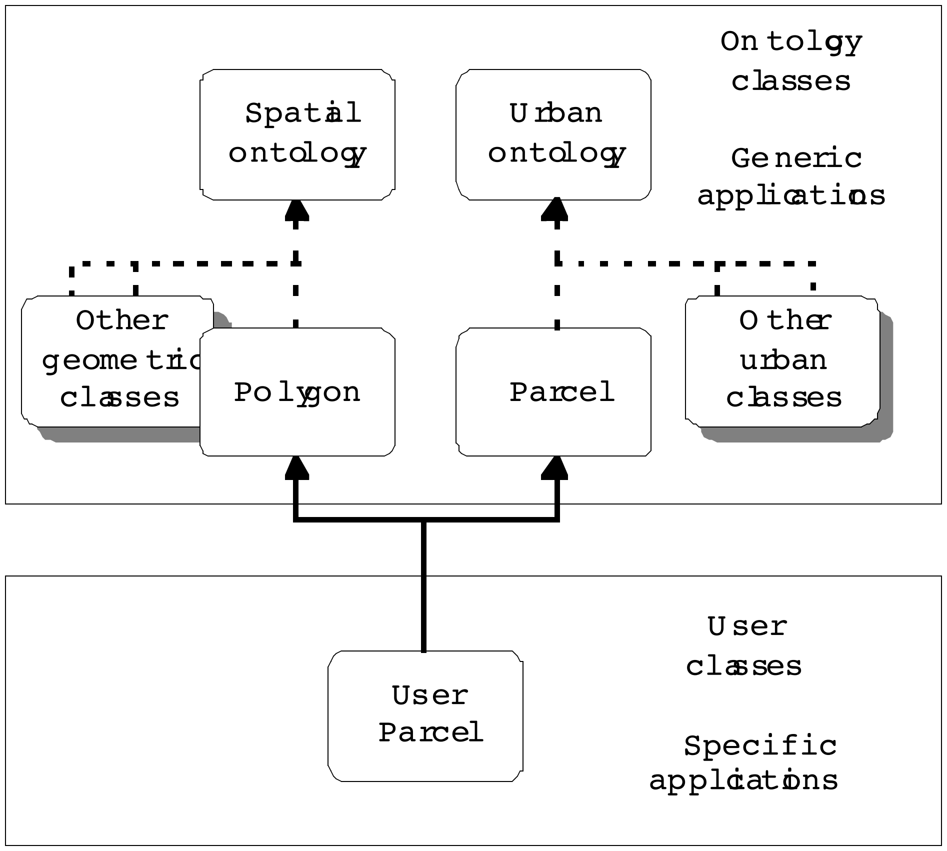

Figure 2 - Parcel class

Consider now the case of an electricity pole. For the utility company, the user class `User Pole` is derived from `Pole` in the urban ontology, from `Geometric Object` in the geometric ontology, and from `node` in the network ontology. For the local government, which is more interested in the existence and the position of the pole than in its role in an electrical network, a `User Pole` class does not need to include a `Network` class parent. This case shows an interesting point on information sharing. Information about poles can be transferred from the utility company to the local government in an easy manner. In this case, the utility company just wants the `Pole` and the `Geometric` interfaces to be publicly accessible. The `Node` interface is considered private and is not exported, either because the local government does not want it or because the utility company considers it private or confidential, and does not want to export it (Figure 3).

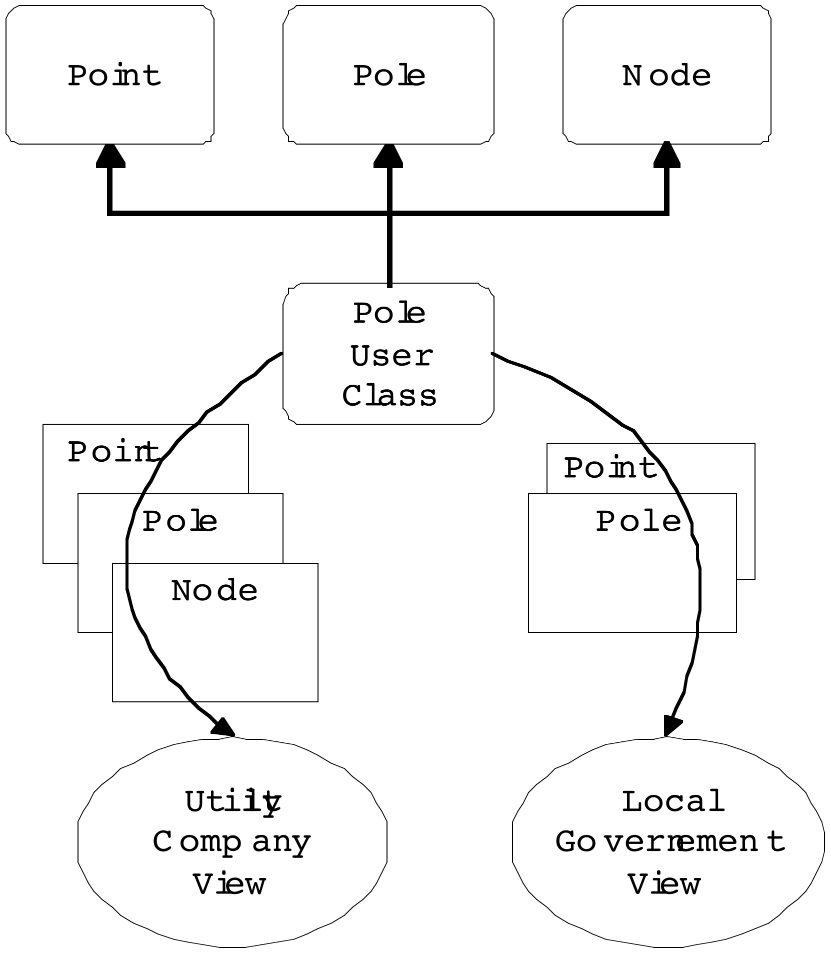

Figure 3 - User class: pole

4.2 Navigation

In general, user classes have to inherit from more than one ontology class. Therefore, the ability to implement multiple inheritance is necessary to build new classes from various ontologies. This is done by using a basic class as a parent for all ontology classes. Like in CORBA (OMG 1991) and Java (Gosling and McGilton 1995), we propose here that all classes should be derived from a basic class, called `Object`. This class has two methods that are fundamental for ontology navigation, the methods `Up()` and `Create_From()`. The method `Up()`, when applied to an object, returns an object of the immediate super class. The method `Create_From()` instantiates a version of the class from an instance of the immediate super class. These two methods provide the means to navigate through the whole ontology tree. Since each class in the ontology tree is derived from the basic class, each interface inherits the necessary navigation tools. So if the navigation methods are applied to the `Network` interface of the `User Pole` class, the class returned is the next upper class in the network ontology tree.

Consider the following example: from the class `Census_Tract`, two subclasses are derived: `Census_Tract_A` and `Census_Tract_B`. The user wants to apply an operation `population_estimate` on `Census_Tract_A`, but the method is available only in `Census_Tract_B`. To do this is necessary to instantiate `Census_Tract_A` as `Census_Tract` using `Up(),` and instantiate `Census_Tract` as `Census_Tract_B` using `Create_From ()` (Figure 4). Then the operation can be applied and the result is available to the user.

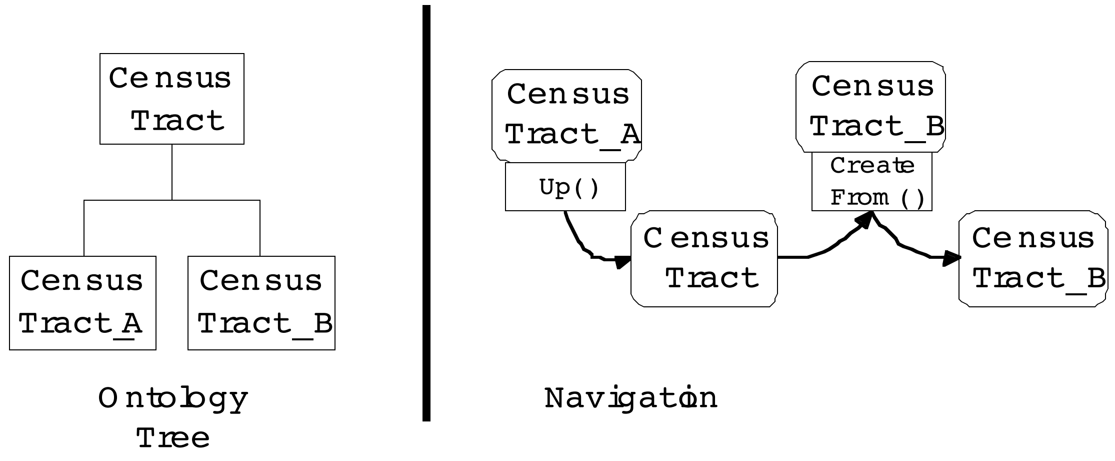

Figure 4: Ontology tree navigation

4.3 Views

A city is a very complex environment and the coexistence of different views of the same entity is usual. These views can exist within the local government where, for instance, the taxes department has to have precise limits of parcels for assessment purposes, while another department only needs a symbol indicating that there exists a parcel or an address in that approximate location. The transportation department can have a different view of the same area, disregarding completely information about parcels, and just focusing on the network transportation links.

Since ODGIS has an object-centered design and supports multiple inheritance it leads to multiple views of the data. An object plays many roles. These roles represent many views to the user. The views are derived from the multiple ontologies and are mapped one by one from them. These views can also be combined, generating new views. We can have a geometric view, a network view, or an alphanumeric relational-like view (Figure 5).

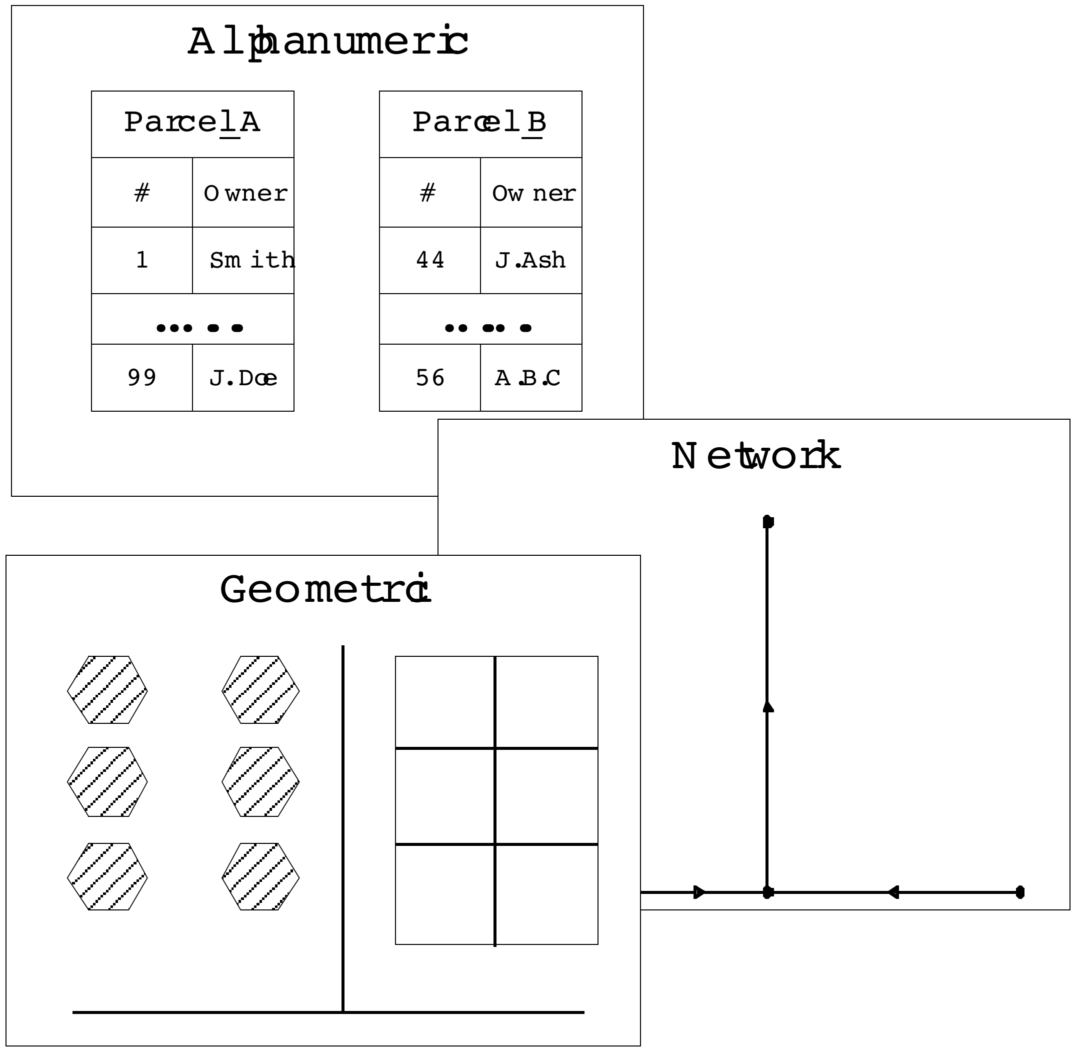

Figure 5: Multiple views

All views of one object are mapped directly onto the original object in the data warehouse. There is only one object, but it can be seen with many faces. We apply here the concept of object identity defined by OMG, in which each object has a unique identity that is distinct from and independent of any of its characteristics. The identity of the object is constant, although its characteristics may vary during its lifetime.

The views can be combined, enabling the user to have a geometric/alphanumeric view. An example of the use of this combined view is a "point-and-click" operation over a parcel that highlights its shape and shows its alphanumeric data.

4.4 Ontology Integration

The complexity of the geographic objects asks for the combination of multiple ontologies. To build a geographic ontology, it is necessary to combine, for instance, a spatial ontology with a geometric ontology and a spatial reference system ontology. We propose here two types of ontology integration. First, before the generation of the translated components, new ontologies can be composed through derivation of existing ontologies or through the insertion of existing ontology references into new ontologies. This integration is done at the top-level, domain, and task ontologies. These ontologies are stable, well developed, and are subject to few updates. At the level of an application ontology, more updates are expected. Applications should be more flexible and evolve with time. We propose that ontology integration at this level should be done at the level of the classes. Through the use of multiple inheritance new classes are created. The new class plays many roles that correspond to the classes used as parents. Each role can come from a different ontology, so the ontology integration is represented by these classes.

5 IMPLEMENTATION ISSUES

A complete architecture for a GIS that uses classes derived from ontologies is shown in (Fonseca and Egenhofer 1999). We give here a brief description of an ODGIS architecture. The main components of the architecture are the ontologies, the container, the data warehouses, and the user interface. The architecture includes a coordinator that integrates all other components. The coordinator is also in charge of finding services on the network, and redirecting them to the components. These services can include an ontology-based search engine for geographic information (Finch and Small 1999), a query language server, or a framework that can function as a user interface with other embedded services. Such a system deals with instances of classes called *objects*. These objects are extracted from geographic databases and carry data and operations. One of the most suitable options for implementing interoperable objects or components (Betz 1994) that need to share both code and data across a heterogeneous network is the use the programming language Java (Clemens 1996; Lewandowsky 1998).

There are two options for implementing Java (Gosling and McGilton 1995) objects from ontologies. First the objects can be generated from ontologies specified in an ontology editor such as Ontolingua, which has the ability to create CORBA IDL headers from ontology components. A CORBA IDL header is a skeleton of ontologies and its components, which should be complemented by implementation code written in Java. The second option is to generate Java interfaces from an ontology editor that has this capability. Since Ontolingua is not able to do this, it is necessary to develop an ontology editor to do this kind of work or implement an extension to Ontolingua enabling the translation of ontologies into Java interfaces.

5.1 Ontology Editor

A group of local government GIS users can create an urban ontology using the Ontolingua editor. The editor allows multiple users to work simultaneously in the specification of the ontology. After that, the user may query and update the ontologies using remote applications on the Internet. The basic ontology for this group of users includes a description of land use parcels. Such a description is generic enough to be used for the whole country. Starting at this point, the ontology can be specified for each state, reflecting specific legislation. From land parcel ontologies at the state level each municipality can derive its own ontology.

Once the ontology is specified, Ontolingua has facilities for translating ontologies from repositories into application environments like CORBA. A CORBA IDL is a skeleton of ontologies and its components, which should be complemented by implementation code written in Java (Gosling and McGilton 1995).

5.2 Java Objects

The creation of user classes from ontology classes is accomplished through inheritance. In object orientation, inheritance is a mechanism where new classes can reuse the implementation of the parent class. New methods and attributes can be added to the parent class, thus characterizing a new class and, at the same time, using the knowledge embedded in the parent class. Inheritance brings another benefit with interface conformance. The interface of the class determines the way the class is seen by the outside world. Subclass interfaces conform to parent interfaces. With this approach, a subclass can be used where a parent class is needed, since the behavior of the subclass is expected to be the same as that of the parent class.

One of the key features of ODGIS is the use of multiple inheritance. Multiple inheritance is not a widely accepted concept, because along with its benefits it brings some problems. One of them arises when methods with the same name are inherited from more than one superclass of a given class. This ambiguity has to be solved by the compiler. Some languages do not accept that, while others precede the original method names with qualifiers, to establish a distinction.

Java does not support multiple inheritance directly. Its basic mechanism for inheritance, using the keyword `extends`, allows a new class to inherit from only one parent class. To support multiple inheritance, Java introduces the concept of *interface*. A Java interface describes the set of public methods that a class that implements the interface must support, and also their calling conventions. But a Java interface does not implement those methods. Each descendant class has to provide the code for each existing interface method. Since new classes can implement more than one interface, multiple inheritance can be achieved in Java (Gosling and McGilton 1995).

In order to use Java to implement the application objects derived from ontology classes, we use a technique called *interface-delegation* (Tempero and Biddle 1998). This mechanism helps the implementation of multiple inheritance in Java, since the language accepts only multiple inheritance of interfaces, but not of classes.

The interface-delegation technique uses a combination of delegation and interface conformance. Delegation is used to simulate the form of parent class reuse (Rumbaugh *et al.* 1991). In delegation (Stein 1987), an object uses a specialized class that is embedded in itself to perform special operations. In this case, the object is composed of other objects, which it uses to perform the methods that it should have inherit, but did not because of the absence of a multiple inheritance implementation. Interface conformance of a subclass to a parent class means that an instance of a subclass can be safely used as a replacement for an instance of the parent class. The set of features of the subclass includes the complete set of the parent class public features.

To achieve the benefits of delegation and interface conformance in Java, the implementation of the classes should be separate from the class definition. The class should be implemented using a Java `interface`, while the implementation is achieved using a Java `class`.

The issues of the interface-delegation technique that remain to be examined are ambiguities and self-calls. Ambiguities should be solved by the designer of the classes, since the Java compiler does not accept implementations of the same method from more than one class, and the delegation wrapper does not allow ambiguities to occur. Self-calls happen when a method calls another method defined in the same class. It is possible that the method lockup uses the object method rather than the class containing the method actually under execution. The only way to avoid this is by copying the code of the implemented method to the delegated method.

The interface-delegation technique allows the use of multiple inheritance in Java without the drawbacks usually associated with languages that have built-in multiple inheritance implementation. The technique allows parent class code reuse and interface conformance.

6 CONCLUSIONS AND FUTURE WORK

This paper proposed the creation of software components from ontologies as a way to share knowledge and information. These software components are derived from ontologies using an object-oriented mapping. The translation of an ontology into an active information system component leads to *ontology-driven geographic information systems* (ODGIS).

The mapping of multiple ontologies to the system classes is achieved through object-oriented techniques using multiple inheritance. This kind of mapping allows partial integration of information when completeness was impossible. This paper has demonstrated that ODGIS can play an important role in enabling information and knowledge sharing.

The possibility of having multiple views of a single geographic object supports the implementation of an object-oriented model of the geographic world. The model used to implement multiple views presented here answers the requisites of view integration (Laurini and Thompson 1992).

We proposed the use of a special parent class that allows navigation from application ontologies to top-level ontologies, passing through domain and task ontologies. This navigation capability shortens the gap between generic and specialized ontologies, enabling the sharing of software components and information. OGDIS employs user classes that are derived through multiple inheritance from various ontologies to solve schematic heterogeneity.

This work requires an ontological commitment from users and information providers. A further study should investigate how to incorporate approaches that allow composition of pre-existing independent developed ontologies, for instance, through the use of a context algebra to compose diverse ontologies (Wiederhold and Jannink 1999) and the matching of synonym, hyponym and hypernym terms (Kashyap and Sheth 1996; Mena *et al.* 1996; Mena *et al.* 1998).

ACKNOWLEDGMENTS

This work was partially supported by the National Science Foundation under NSF grant numbers SBR-9700465 and IIS-9970123. Max Egenhofer's research is further supported by NSF grants IRI-9613646, BDI-9723873, and EIA-9876707; by the National Imagery and Mapping Agency under grant number NMA202-97-1-1023; by the National Institute of Environmental Health Sciences, NIH, under grant number 1 R 01 ES09816-01; by a contract with Lockheed Martin; and by Bangor Hydro-Electric Co.

REFERENCES

P. Alexandroff (1961) *Elementary Concepts of Topology*. Dover Publications, Inc., New York.

D. Arctur, D. Hair, G. Timson, E. Martin, and R. Fegeas (1998) Issues and Prospects for the Next Generation of the Spatial Data Transfer Standard (SDTS). *International Journal of Geographical information Science* 12(4): 403-425.

S. Bergamaschi, S. Castano, S. Vermercati, S. Montanari, and M. Vincini (1998) An Intelligent Approach to Information Integration. in: N. Guarino (Ed.), *Formal Ontology in Information Systems*. IOS Press, Amsterdan, Netherlands.

M. Betz (1994) Interoperable Objects. Dr. Dobb's Journal 4(220):22-26.

Y. Bishr (1997) *Semantic Aspect of Interoperable GIS*. Ph.D., Wageningen Agricultural University, The Netherlands.

Y. Bishr (1998) Overcoming the Semantic and Other Barriers to GIS Interoperability. *International Journal of Geographical Information Science* 12(4): 299-314.

Y. Bishr, H. Pundt, W. Kuhn, and M. Rdwan (1999) Probing the Concepts of Information Communities - A First Step Toward Semantic Interoperability. in: M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman (Eds.), *Interoperating Geographic Information Systems*. pp. 165-180, Kluwer, Norwell, MA.

L. Cardelli (1984) A Semantics of Multiple Inheritance. in: G. Kahn, D. McQueen, and G. Plotkin (Eds.), *Semantics of Data Types*. *Lecture Notes in Computer Science* 173, pp. 51-67, Springer-Verlag, New York.

R. Chin and S. Chanson (1991) Distributed, Object-Based Programming Systems. *ACM Computing Surveys* 23(1): 91-127.

P. Clemens (1996) *Coming Attractions in Software Archictecture*. Carnegie Mellon University, Pittsburgh, PA, Technical Report CMU/SEI-96-TR-008.

W. Clocksin and C. Mellish (1981) *Programming in Prolog*. Springer-Verlag, New York.

M. Egenhofer and A. Frank (1992) Object-Oriented Modeling for GIS. *Journal of the Urban and Regional Information Systems Association* 4(2): 3-19.

A. Farquhar, R. Fikes, and J. Rice (1996) *The Ontolingua Server: a Tool for Collaborative Ontology Construction*. Knowledge Systems Laboratory - Stanford University, Stanford, CA, Technical Report KSL 96-26.

I. Finch and E. Small (1999) Information Brokers for a WEB-based Geographic Information System. in: M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman (Eds.), *Interoperating Geographic Information Systems*. pp. 165-180, Kluwer, Norwell, MA.

F. Fonseca and M. Egenhofer (1999) Ontology-Driven Geographic Information Systems. in: *7th ACM Symposium on Advances in Geographic Information Systems*, Kansas City, MO, pp. 14-19.

A. Frank (1997) Spatial Ontology: A Geographical Point of View. in: O. Stock (Ed.), *Spatial and Temporal Reasoning*. pp. 135-153, Kluwer, Dordrecht, The Netherlands.

M. Genesereth (1990) *The Epikit Manual*. Epistemics, Inc., Palo Alto, CA, Technical Report.

M. Genesereth and R. Fikes (1992) *Knowledge Interchange Format*. Stanford University, Knowledge Systems Laboratory, Technical Report KSL-92-86.

M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman (1999) *Interoperating Geographic Information Systems*. Kluwer, Norwel, MA.

J. Gosling and H. McGilton (1995) *The Java Language Environment: a White Paper*. Sun Microsystems, Mountain View, CA, Technical Report.

T. Gruber (1991) The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases. in: *Principles of Knowledge Representation and Reasoning*, Cambridge, MA, pp.601-602.

T. Gruber (1992) *A Translation Approach to Portable Ontology Specifications*. Knowledge Systems Laboratory - Stanford University, Stanford, CA, Technical Report KSL 92-71.

N. Guarino (1998) Formal Ontology and Information Systems. in: N. Guarino (Ed.), *Formal Ontology in Information Systems*. pp. 3-15, IOS Press, Amsterdam, The Netherlands.

W. Huxhold (1991) *An Introduction to Urban Geographic Information Systems*. Oxford University Press, New York.

V. Kashyap and A. Sheth (1996) Semantic Heterogeneity in Global Information System: The Role of Metadata, Context and Ontologies. in: M. Papazoglou and G. Schlageter (Eds.), *Cooperative Information Systems: Current Trends and Directions*. pp. 139-178, Academic Press, London.

K. Kemp and A. Vckovski (1998) Towards an Ontology of Fields. in: *Third International Conference on GeoComputation*, http://www.geog.port.ac.uk/geocomp/geo98/ Bristol, UK.

W. Kent (1994) Object Orientation and Interoperability. Hewlett-Packard Laboratories, Palo Alto, CA, Technical Report HPL-93-58.

R. Laurini and D. Thompson (1992) *Fundamentals of Spatial Information Systems*. Academic Press, London.

S. Lewandowsky (1998) Frameworks for Component-Based Client/Server Computing. *ACM Computing Surveys* 30(1): 4-27.

L. McKee and K. Buehler, Eds. (1996) *The Open GIS Guide*. Open GIS Consortium, Inc, Wayland, MA.

E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth (1998) Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure. in: N. Guarino (Ed.), *Formal Ontology in Information Systems*. pp. 269-283, IOS Press, Amsterdam, The Netherlands.

E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi (1996) OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. in: *First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, pp. 14-25, Brussels, Belgium.

G. Miller (1995) WordNet: A Lexical Database for English. *Communications of the ACM* 38(11): 39-41.

R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swartout (1991) Enabling Technology for Knowledge Sharing. *AI Magazine* 12(3): 16-36.

J. Nunes (1991) Geographic Space as a Set of Concrete Geographical Entities. in: D. Mark and A. Frank (Eds.), *Cognitive and Linguistic Aspects of Geographic Space*. pp. 9-33, Kluwer, Dordrecht, The Netherlands.

OMG, Ed. (1991) The Common Object Request Broker: Architecture and Specification, Revision1.1. OMG Document No. 91.12.1 Framingham, MA.

Y. Papakonstantinou, H. Garcia-Molina, and J. Widom (1995) Object Exchange Across Heterogeneous Information Sources. in: *IEEE International Conference on Data Engineering*, Taipei, Taiwan, pp. 251-260.

A. Rodriguez, M. Egenhofer, and R. Rugg (1999) Assessing Semantic Similarity Among Geospatial Feature Class Definitions. in: A. Vckovski, K. Brassel, and H.-J. Schek (Ed.), *Interoperating Geographic Information Systems - Second International Conference, INTEROP'99, Zurich, Switzerland*. *Lecture Notes in Computer Science* 1580, pp. 1-16, Springer-Verlag, Berlin.

J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen (1991) *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, NJ.

A. Sheth (1999) Changing Focus on Interoperability in Information Systems. in: M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman (Ed.), *Interoperating Geographic Information Systems*. pp. 165-180, Kluwer Academic Publishers, Norwell, MA.

A. Sheth and J. Larson (1990) Federated Databases Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys* 22(3): 183-236.

B. Smith (1993) Ontology and the Logistic Analysis of Reality. in: N. Guarino and R. Poli (Eds.), *International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation*, Institute for Systems Theory and Biomedical Engineering of the Italian National Research Council, Padua, Italy, pp. 51-68.

B. Smith (1995) On Drawing Lines on a Map. in: A. Frank and W. Kuhn (Eds.), *Spatial Information Theory - COSIT 95*, Vienna, Austria, *Lecture Notes in Computer Science* 988, pp 475–484, Springer-Verlag, Berlin.

B. Smith (1998) An Introduction to Ontology. in: D. Peuquet, B. Smith, and B. Brogaard (Eds.), *The Ontology of Fields.* Technical Report, National Center for Geographic Information and Analysis, University of California, Santa Barbara, pp. 10-14.

B. Smith and D. Mark (1998) Ontology and Geographic Kinds. in: T. Peucker and N. Chrisman (Eds.), *International Symposium on Spatial Data Handling*, Vancouver, Canada, pp. 308-320.

L. Stein (1987) Delegation is Inheritance. in: *OOPSLA'87*, Orlando, FL, pp 138-146.

E. Tempero and R. Biddle (1998) *Simulating Multiple Inheritance in Java*. Victoria University of Wellington, School of Mathematical and Computing Sciences, Wellington, New Zealand, Technical Report CS-TR-98/1.

A. Vckovski (1998) *International Journal of Geographical Information Science - Special Issue: Interoperability in GIS*. 12(4): 297-298.

A. Vckovski, K. Brassel, and H.-J. Schek (1999) Preface. in: A. Vckovski, K. Brassel, and H.-J. Schek (Eds.), *Interoperating Geographic Information Systems-Second International Conference, INTEROP'99, Zurich, Switzerland*, Zurich, Switzerland, *Lecture Notes in Computer Science* 1580, Springer-Verlag, Berlin.

G. Wiederhold (1991) *Mediators in the Architecture of Future Information Systems*. Stanford University, Technical Report.

G. Wiederhold (1994) Interoperation, Mediation and Ontologies. in: *International Symposium on Fifth Generation Computer Systems (FGCS94)*, Tokyo, Japan, pp. 33-48.

G. Wiederhold and J. Jannink (1999) Composing Diverse Ontologies. Stanford University, Scalable Knowledge Composition (SKC) Project, Technical Report.

M. Worboys (1994) Object-oriented Approaches to Geo-Referenced Information. *International Journal of Geographical Information Systems* 8(4): 385-399.