Ownership Semantics in the Financial Domain

J. Geller, C. Rush, L. Liu, M. Halper and Y. Perl
Department of Computer and Information Sciences
New Jersey Institute of Technology
Newark, NJ 07102

Abstract
This paper examines the nature and characteristics of the ownership relationship between objects in an object-oriented database. The ownership relationship is distinguished from the Part-Whole relationship since these two are often confused. Dimensions of the ownership relationship are detailed and described with clarifying examples from the financial securities markets. These dimensions are: 1) exclusiveness, 2) dependency, 3) limitation, 4) status, 5) settlement, 6) documentation and 7) inheritance. Formal definitions of the dimensions of the ownership relationship are sketched out for the purpose of extending current Object-Oriented Database Models (OODBs).

Keywords
Ownership, Semantic Relationships, OODB, Object Modeling.

1. INTRODUCTION

A great deal of effort has been expended in the literature on analyzing and formalizing the semantics of the IS-A and part-of relationships. The IS-A relationship is the basis for generalization and specialization and allows reasoning by inheritance and transitive closure about the fundamental properties of object classes. The part-of relationship is fundamental to our understanding of the physical world. However, most other relationships have not been analyzed in the literature to the same depth.

We feel strongly that in the area of e-Commerce (and commerce in general) the central relationship is neither IS-A nor part-of but ownership. None of the frameworks that deal with relationships on an abstract level can capture the intricacies of ownership, while known models of IS-A and part-of cannot be automatically transferred to ownership. Ownership may be defined as a relationship between a legal entity or a natural person (the "owner") and an owned object, such that the owner has some defined rights to use an attribute(s) of the owned object for the owner's self-determined purposes. Ownership can have different operational characteristics (for example, transferable versus non-transferable) depending on the instrument owned and is often defined by statute, securities regulations, and the court of law.

In all commercial activities a central data managing task is to maintain correct information about the ownership of objects, and about the orderly, rule and law governed transfer of ownership (ownership transaction) of these objects. With the advent of modern financial instruments, these transfers have become bewilderingly complex, and e-Commerce will greatly benefit from a theoretical model of ownership and ownership transfer. In other words, we feel that with the ascent of e-Commerce, ownership deserves the same degree of attention as IS-A and part-of.

In e-Commerce, transactions need to be checked and performed electronically. One cannot perform a transaction without checking some features of an ownership relationship. For example, a person or a company cannot transfer ownership of an object without having exclusive ownership rights in it. Hence, if an object is owned by partners, there is a need to check the nature of the partnership or obtain a consent from a partner. Hence there is the need to model the dimension of exclusiveness in ownership. The ownership of some objects like cars needs to be registered to enable transactions of ownership. Some transactions take a period of time till they are completed, for example the sales of stocks. Hence various dimensions of ownership, discussed in this paper need to be accurately modeled and electronically accessible to enable e-Commerce transactions.

With this great practical importance of the ownership relationship, our goal is to accurately capture it in an object-oriented database (OODB) model. Ownership inherently provides owners with unique powers and abilities and the object-oriented model provides an appropriate and efficient choice for modeling relationship characteristics. All current OODBs would create an ordinary relationship between two classes that is merely titled "owns" and would provide no unique meaning for the ownership relationship. The characteristics of the ownership relationship itself would have to be coded into the rules that define the database application to include realistic operational meaning for ownership.

This paper extends a model previously developed for providing meaning to the Part-Whole relationship in an OODB. It distinguishes the ownership relationship from other semantic relationships and defines a set of characteristics of the ownership relationship's interactions and functionality. This set of characteristics will be referred to as *dimensions*. We define the dimensions of the ownership relationship such that important distinctions between several kinds of ownership relationships can be automatically enforced by a database system. Once the relationship is determined to exist between two objects, the values of the dimensions will be set by the database modeler at schema design time and the database system will enforce them. Hence, the abilities and limits that define the relationship between owner and property will be uniquely modeled within the OODB.

Our ownership model incorporates some dimensions used in the Part-Whole model [12, 11, 10, 9, 20] and adds others to fully characterize the ownership relationship. There are seven dimensions: 1) exclusiveness, 2) dependency, 3) limitation, 4) status, 5) settlement, 6) documentation, and 7) inheritance. In this paper, we focus on the preliminary issue of how to better understand the ownership relationship by distinguishing it from similar relationships, especially part-of.

The remainder of this paper is organized as follows: Section 2 is a survey of previous work. Section 3 discusses a model of ownership for an OODB. Section 4 is a classification methodology for ownership relationships. Section 5 describes the distinction between part-of and ownership in a business context. Section 6 contains a brief overview of the dimensions of ownership. Section 7 reviews a prototype implementation. Section 8 contains conclusions.

2. PREVIOUS WORK

Papers related to ownership modeling in OODBs come from a variety of active research areas, but no one covers all aspects of the ownership model presented here. Ownership and semantic relationships have been discussed in the following sub-areas of computer science: legal modeling, extensions to database and object models, and definition of aggregate and composite relationships. We will consider each of these areas and their relevance to our current research in this section.

L. Thorne McCarty has written extensively on the ownership relationship [17]. He presents a description of the legal groundwork for ownership and presents his Language for Legal Discourse ("LLD") to model that relationship. In [18], McCarty uses his LLD to model a corporate tax case in America. While the theme of [18] is to implement the legal logic of the case in McCarty's TAXMAN II theory, the article does give an example of the ownership relationship modeled in LLD. LLD is implemented in PROLOG. McCarty's modeling of legal reasoning has played an important foundational role in efforts to model legal concepts in computer applications [16, 19, 21, 25].

Another area of related work is research on extending database models to incorporate relationship semantics. We have grouped these approaches into four areas: Relational Database Management Systems

(RDBMS), Object-Relational Database Management Systems (ORDBMS), Object-Oriented Database Management Systems (OODBS) and the ODMG Object Model [3]. The extensions to RDBMSs generally include modeling relationships as tables, and further extending SQL-like languages to incorporate relationships and inheritance hierarchies pertaining to those relationships [23, 26]. In [30] an attempt was made to implement semantic extensions to an ORDBMS; however, it was concluded that the current approaches taken to extend ORDBMSs do not support precise specifications or automatic maintenance of semantic relationships.

Our paper discusses the value of extending existing OODBSs with the ownership relationship. OODBSs provide the most likely candidate to accommodate the changes necessary to support semantic relationships. Although there has been little work on the OODB's ownership relationship, other than our own [29], there has been similar work on generic relationships, Part-Whole relationships, and dynamic relationships in OODBSs. These approaches support our contention that the most easily and accurately extensible data models are object-oriented [2, 4, 12, 20].

In [24], the idea is introduced that a complex relationship modeled in an OODB needs to be dynamic over its lifetime. This idea may be incorporated into our model by the introduction of a kind of schema evolution. The general topic of schema evolution in the representation of semantic relationships of OODBs is a subject of future investigation.

We have found that there remains considerable discussion and debate in the literature on extensions to the object model itself and the definitions of aggregate and composite relationships [3, 5, 6, 7, 13, 22]. A paper by Henderson-Sellers and Barbier [14] alludes to the continued difficulty in defining aggregation. They argue that ownership is a consequent property of a set of primary aggregation properties that they define. Their classification of the ownership relationship may work well within the tradition of software engineering and data structures. However, our view of ownership is based on real world examples of ownership. Hence, we will show in this paper that the ownership relationship is a separate and distinct relationship which can be systematically separated from Part-Whole and other relationships.

In our previous work on the semantics of parts we drew heavily on a seminal paper by Winston *et al.* [27]. Their paper distinguishes between six different senses of the part relationship. In this current paper we are again drawing on ideas of Winston *et al.* on how to distinguish between different kinds of one semantic relationship. Another approach that characterizes the semantics of relationships based on primitive dimensions is [15].

3. MODELING OWNERSHIP IN AN OODB

In order to represent the meaning of ownership in an OODB, we define the ownership relationship as a relationship between two classes, at least one of which must model a legal entity or natural person (the "owner class") such that instances of the owner class have some defined rights to use an attribute(s) of instances of the owned class for the owner's self-determined purposes.

Objects that can own something will be modeled by an owner class and will be limited to natural persons or legal entities, as provided and practiced in the law. Further, we limit the entities that can be owned to those objects for which some right(s) can be assigned from the object itself to the owner as discussed later. The class of owned objects can be extensive and diverse, including real property, intellectual property, various financial products, etc.

As we showed in our previous work on the part relationship, OODBs make it comparatively easy to add new semantic relationships to an existing database system. In addition, there is real world evidence that complex financial applications can benefit from OODBSs when compared to traditional relational databases [1]. The ownership relationship can be represented in an OODB system, using the approach that we applied to part-of. This provides a framework for implementing the ownership relationship in an OODB in a reusable and automatic fashion. Once implemented in the OODB model, any ownership relationship can be defined by its dimensions and its use will be enforced by the database management system automatically, thus avoiding the repeated rewriting and reimplementation of the ownership relationship as part of different applications.

Figure 1 shows an example of an object-oriented schema that has been augmented by ownership links. According to this schema, a person can own different kinds of objects. Every owned object class points with a dashed arrow to the owner class. In the figure, we additionally annotated all ownership relationships with numbers from 1 to 7.

4. OWNERSHIP CLASSIFICATION METHODOLOGY

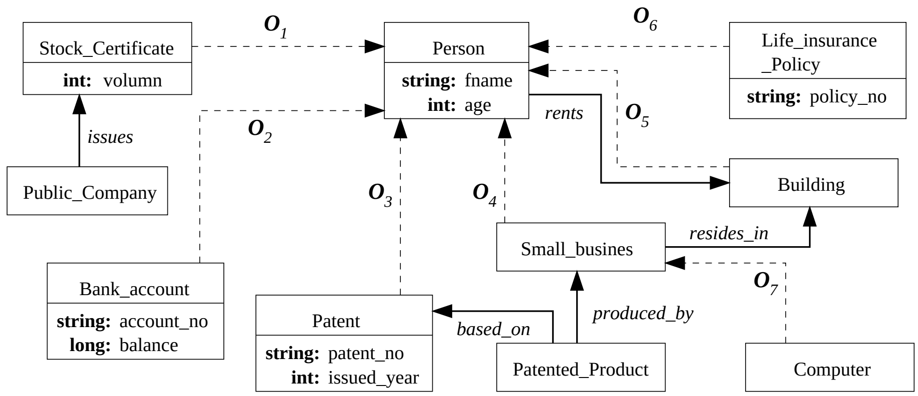

FIGURE 1. An OODB Ownership Schema

We now present a six-step methodology we have developed to fully distinguish the ownership relationship from other relationships (especially part-of).

1. Translate a relationship sentence ("The car has wheels.") into one that connects two objects with the word "owns." If necessary, switch the owner object with the owned object to produce the best English ownership sentence possible. For instance, "The wheels own the car." is switched to, "The car owns the wheels."
2. Reject the possibility of an ownership relationship between the two objects if the best possible sentence is not natural in English.
3. If not rejected via Step 2, reject the possibility of an ownership relationship between the two objects if the best possible sentence changes the meaning of the original relationship, according to the judgement of the database designer. (For an example, see the last entry in Table 1).
4. If not rejected via Step 3, reject the possibility of an ownership relationship if the owner is not eligible for an ownership relationship. Only legal entities and natural persons are eligible owners. We call this the *eligibility* criterion.[1]
5. If not rejected via Step 4, reject the ownership relationship if there is no evidence of a legal right in the owned object. We call this is the *evaluation* criterion.
6. If not rejected at this point, the relationship is classified as an ownership relationship.

[1]We are not interested in modeling the few isolated cases where an animal may own property, e.g., by being included in the will of its deceased owner.

It is necessary to consider both the usage and meaning of the translated sentence (Steps 2 and 3) in the classification process. A translated sentence can be correct English even though its meaning can change. Any change in meaning implies that the translation is not equivalent, e.g., "has" or "part-of" is not the same as "owns." To illustrate the importance of examining the usage and meaning of the translated sentence, we consider the six kinds of parts (Table 1) as proposed by Winston *et al.* [27]. These examples were all included in [27] and provide a basis for our observations.

Table 1 shows an example for each sense of the part-of relationship and a corresponding attempt to translate each example into an ownership sentence. The last column evaluates the result of each translation. All the examples are rejected as possible ownership relationships and, in most cases, the translation is rejected based on English usage. For example, consider what happens if the component-integral object example "handle is part of a cup" is translated to "a cup owns a handle." This translation is clearly rejected as not natural in English. From the examples in Table 1, only the last case can not be rejected based on English usage; rather, it is rejected because it changes the original meaning (Step 3) from one of physical location to one of possession.

Returning to the previous example, "Joe has a car," Steps 3 through 6 can now be used to fully classify the relationship. The meaning of "Joe owns a car" and "Joe has a car" are similar as both suggest possession (Step 3). Joe is a natural person and is an eligible owner and hence passes the *eligibility* criterion (Step 4). Let us assume that the database modeler has the evidence that Joe is the title holder of the car and thus passes the *evaluation* criterion (Step 5). Since the ownership relationship has not been rejected at this point, the relationship is accepted as one of ownership.

In an object-oriented environment it is, of course, not sufficient to decide that one specific "Joe" owns one specific "car." Rather, this judgement has to be extended to the most appropriate class that "Joe" is an instance of, and the most appropriate class that "car" is an instance of. Thus, the data modeler can conclude that an ownership relationship holds between a class such as "adult person" and a class such as "vehicle." Instances of both these classes may then be connected by instances of the ownership relationship during use of the database.

5. OWNERSHIP IN THE BUSINESS CONTEXT

TABLE 1. Translation from "Part-of" to "Owns"

Type	"Part of"	"Owns"	Analyze
Component-Integral	A handle is part of a cup. A punchline is part of a joke.	A cup owns its handle. A joke owns its punchline.	Rejected usage. Rejected usage
Member-Collection	A tree is part of a forest. This ship is part of a fleet.	The forest owns its trees. A fleet owns its ships.	Rejected usage. Rejected usage.
Portion-Mass	This slice is part of a pie. A yard is part of a mile.	A pie owns its slices. A mile owns its yards.	Rejected usage. Rejected usage.
Stuff-Object	A martini is partly alcohol. Water is partly hydrogen.	Alcohol owns a martini. Hydrogen owns water.	Rejected usage. Rejected usage.
Feature-Activity	Paying is part of shopping. Bidding is part of playing bridge.	Shopping owns paying. Playing bridge owns bidding.	Rejected usage. Rejected usage.
Place-Area	The Everglades are part of Florida. An oasis is part of a desert.	Florida owns the Everglades. A desert owns an oasis.	Rejected meaning. Rejected usage.

In the above examples, the part-of relationships are easily distinguished from ownership relationships. We have found that in order to be in an ownership relationship, the owned object must be separable from the owner, i.e., the owner must be able to sell or transfer the rights in the owned object without leaving the owner less whole, or defective. In addition, an owner cannot be indistinguishable from its owned object, i.e., a pie does not have an ownership relationship with its slices. Indeed, based on our observations and the classification of Winston *et al.*, the member/collection type of part analysis (the second entry in Table 1) is the only part relationship that provides some interesting examples, which can, potentially, be classified as ownership relationships.[2]

We propose that since the part in the member-collection relation type is separable from the whole, and is not functionally connected or similar in composition to the whole, the relation can be reconsidered to be one

[2]Winston *et al.* present a table where each part relation is classified by its elements: functional, homeomerous, and separable. The member-collection relation is the only relation that exhibits this combination of characteristics.

of ownership rather than one of Part-Whole. The particular combination of separable elements that are (i) not functionally connected and (ii) not indistinguishable from the whole, only exists in the member-collection sense of the Part-Whole relation [27].

We conclude that this unique combination of characteristics (namely separable and distinguishable) is a necessary requirement for any relationship to be classified as an ownership relationship. Thus, these characteristics provide conditions that enable reclassification of a "has a" or a "part-of" relationship into an ownership relationship.

The following example from the business world demonstrates a relationship that can be classified as ownership relationship. Consider the statement, "NBC is part of GE" and its translation, "GE owns NBC." At first glance, "NBC is part of GE" could be classified by a data modeler as a Part-Whole relationship. The first sentence implies that NBC is part of the collection of things GE owns. The translation means essentially the same thing. In this case, the translation cannot be rejected based on usage or meaning (Steps 2 and 3). Hence, the next step for the database modeler is to examine the eligibility criterion (Step 4). Is the owner in the translated sentence a legal entity or a natural person? GE is a corporation and hence is a legal entity and is eligible for an ownership relationship. The fifth step, *evaluation*, is then applied by the database designer as follows. Does GE have a legal right in an attribute of NBC? While this is a factual and legal evaluation, we will assume that the database modeler knows that GE does indeed have such an interest. Since the result of applying the evaluation criterion was positive, we propose that "NBC is part of GE" is most accurately classified as an ownership relationship, with GE as the owner and NBC as the owned object. Alternatively, if GE were a legal entity, but with no rights in NBC, then the modeler would classify the relationship as a Part-Whole relationship.

At the level of object-oriented data modeling, it is again necessary to generalize from such an example to a statement about classes and relationships which can be expressed with an object-oriented data definition language. In our case, as NBC and GE are both companies, we can model the ownership by a class "COMPANY" which has a semantic relationships "owns" to itself.

6. DIMENSIONS OF OWNERSHIP

Once the database modeler determines that an ownership relationship exists between two objects, the characteristics of the relationship must be defined. As mentioned in the introduction of this paper, the ownership relationship and allowable transactions vary widely for different ownership relationships and different owned objects. This is especially true for financial instruments that have very specific rules and regulations dictating their terms of ownership.

We refer to the characteristics of the ownership relationship as *characteristic dimensions* or, in short, *dimensions*. An intuitive example of a dimension is whether an ownership is shared or exclusive. Shared and exclusive are values along the exclusiveness dimension. As an example that requires a complicated dimension, consider the fact that all options have fixed expiration dates of 11:59 pm ET on the Saturday following the third Friday of the month. Any customer who wishes to call (buy) or put (sell) his option has to submit notification to his dealer/broker of his desire prior to 5:30 pm ET on the day before the expiration of the option. Failure to provide timely notification results in the option expiring as worthless [8]. Stocks, options, and other financial instruments provide a rich source of examples for the ownership relationship characteristics. We have found that by considering business and financial applications, many of the characteristics of the ownership relationship will be revealed.

The dimensions specified must be sufficient to describe all the relationships and to support all of the different permitted operations for many types of owned objects. These operations include, among others, the establishment and dissolution of ownership relationships between instances of classes between which ownership has been defined, a variety of queries, as well as certain operations that rely on property inheritance. For example, the monetary value of an instance of the class NAMEDFUND would be defined as the sum of the values of the different stocks owned by this fund. The instance of the class NAMEDFUND would inherit the values from the instances of the class STOCK that it is connected to by instances of the ownership relationship.

6.1. Formal Definition of the Ownership Relationship

The formal definition of the ownership relationship in this paper will be kept brief. Preliminary, versions have appeared in [29, 28]. First a generic ownership relationship is defined. Let class B be an owned class and class A be an owner class and the relation between them is Ω_B^A, from *E(B)* to *E(A)* where *E(B)* denotes the extension of the class B. The pair(b,a)$\in \Omega_B^A$, means that the instance *b* of class B is owned by the instance *a* of class A [28]. The ownership relationship between an owner class A and a property class B, $O_{B,A}$ is defined by the following octuplet:

$$O_{B,A} = \langle \Omega_B^A; \chi, \delta, (\mathrm{A}, \mathrm{T}, \mathrm{V}, \mathrm{U}, \Psi), \sigma, \gamma, \beta, (\Upsilon, \Delta, \Phi) \rangle.$$

Based on our review of various owned objects including stocks and bonds, we found seven characteristics dimensions of the ownership relationship. The names and domains of the dimensions are listed below:

$$\begin{array}{rcl}
\textit{Exclusiveness: } \chi & \in & \mathrm{X} = \{\text{exclusive, free-joint,} \\
& & \text{percentage-joint,} \\
& & \text{shared-shares}\}, \\
\textit{Dependency: } \delta & \in & \mathrm{D} = \{\text{owner-to-property,} \\
& & \text{property-to-owner, nil}\}, \\
\textit{Limitation: } (\mathrm{A}, \mathrm{T}, \mathrm{V}, \mathrm{U}, \Psi) & \in & \mathrm{L} = \{\text{action-limited, time-limited,} \\
& & \text{value-limited, unlimited,} \\
& & \text{function-limited}\}, \\
\textit{Status: } \sigma & \in & \mathrm{S} = \{\text{status level, nil}\}, \\
\textit{Settlement: } \gamma & \in & \mathrm{C} = \{\text{confirmed-yes, confirmed-no}\}, \\
\textit{Documentation: } \beta & \in & \mathrm{B} = \{\text{documented, undocumented}\}, \\
\textit{Inheritance: } (\Upsilon, \Delta, \Phi) & \in & \mathrm{V} = \{\text{upSet, downSet, function}\}.
\end{array}$$

6.2. The Exclusiveness Dimension

One way to classify the ownership relationship is to consider whether ownership is exclusive to one owner, joint between two, or several owners, or shared with many owners as in the case of stocks. The set of values for this dimension is as follows:

Exclusiveness: $\chi \in \mathrm{X} = \{$exclusive, free-joint, percentage-joint, shared-shares$\}$,

An Object-Oriented Database system needs to enforce values along this dimension. For instance, it needs to enforce a total of 100% for percentage-joint ownership.

6.3. The Dependency Dimension

The dependency dimension of the ownership relationship describes the fact that, in some cases, the ownership relationship between two objects determines the lifetime of the objects themselves. In other words, the deletion of an object may propagate along the ownership relationship to another object. It appears that at first glance there is no reason to expect that just because one of the objects, or indeed the ownership relationship itself, ceases to exist, the other object in the relationship should also be deleted. However, consider the following examples of owners and financial options.

Briefly stated, an option is the right to buy or sell a particular stock at a preset time and price [8]. The contract an investor enters is classified as either a call or a put for stock at the set price. A put is the right to sell the underlying stock at the set price. If the option is exercised, the owners of both the call and the put are obligated to the transaction at the set price [8].

For the dependency dimension discussion, assume there is an ownership relationship between an investor and a particular option. Assume further that the investor is modeled as an option-investor, a specialized subclass of an investor. In reality, in order for an investor to be able to buy options, his account must be approved by a Registered Options Principal (ROP) prior to the execution of any options [8]. It is therefore likely that an option investor would be modeled as a subclass of the investor class with special characteristics. For example, if the option-investor allows his only option to expire, i.e., the option fails to exists because the date expires, then, assuming the schema just described, the option-investor object may also be deleted from the set of active option-investors since the investor no longer owns any options. This is called an *owner-to-property dependency.* This way of modeling, where information about the data, the investor in this case, is shifted from the data to the database schema, is described in [28].

The formal definition of the Dependency dimension includes this set of values:

Dependency: $\delta \in \mathrm{D} = \{$*owner-to-property*, *property-to-owner*, *nil*$\}$

6.4. The Limitation Dimension

The limitation dimension is a multi-faceted dimension with three major divisions; action, time and value limitations. Action limitations describe a restriction in the particular rights to perform certain operations generally available in the ownership relationship. Time limits describe restrictions in the ability to exercise ownership rights based on time constraints. The (monetary) value limit describes restrictions on the value or price at the time the owner exercises certain rights, such as the price to buy or sell an option. (Only in this section we use the phrases "monetary value" and "dimension value" to distinguish between them.) Combinations of the dimension values will restrict some ownership relationships. The limitation dimension is specified by the following set of dimension values:

Limitation: $\alpha \in$ L =
{*action-limited, time-limited, value-limited, unlimited, function-limited*}

6.5. The Status Dimension

The status dimension allows for differential treatment for different owner classifications. This recognizes that some applications may have different classes of owners that have prioritized access to objects and operations, e.g. corporate resources and fee structures, just to mention a few. It is expected that different applications will have different numbers of owner classifications and different classification rules. Hence this dimension allows for a user-defined number of levels including none (nil). While most applications are not expected to differentiate between owner levels, this dimension is included to enable the distinction in the desired cases.

The dimension is formally defined as:

Status: $\sigma \in$ S = {*status-level-1,status-level-2,....status-level-n, nil*}

6.6. The Settlement Dimension

This dimension is a Boolean value that reflects whether an object's ownership has been transferred from one owner to another.

The formal definition of the settlement dimension is:

Settlement: $\gamma \in$ C = {*confirmed-yes, confirmed-no*}

6.7. The Documentation Dimension

Documented ownership has supporting legal documentation, such as copies of titles, purchase agreements, bills of sales, *etc.*, while undocumented ownership does not. The values for the documentation dimension are as follows:

Documentation: $\beta \in$ B = {*documented, undocumented*}

6.8. The Inheritance Dimension and Derived Attributes

In the ownership relationship, as in other hierarchical semantic relationships, inheritance plays an important and powerful role. Inheritance is used to represent the fact that at least selective attributes can be attained by one class from another via the hierarchical relationship structure [12]. For the ownership relationship, inheritance is selective, upward and downward. Not all the attributes of the owner class instance in an ownership relationship are transmitted downward to its property class instance. Nor are all of the attributes of the property class instance associated with an ownership relationship transmitted upward to the owner class instance. For example, employees of a branch office do not inherit the employees of corporate headquarters. As for the Part-Whole relationship, inheritance of a property (up or down) will have to be explicitly specified by the database designer [12].

Inheritance in an ownership model allows us to express operations that are fairly complicated to implement but easy to express in English by a *declarative* formalism. For instance, by inheritance it is possible to retrieve the "net worth" of a company by adding up the values of all its holdings, without programming this complicated retrieval. This goes beyond what is offered by an SQL-like query language (e.g., OQL). Query languages allow retrieval of information by specifying *what the user wants*, but not *how to get it*. In the case of Object-Oriented modeling of semantic relationships the database designer says *what is true in the world* and the user says what he wants. However, by having a complex ownership model built into the schema of each application, the user can express his query in our example by retrieving the value of a single derived attribute. The whole complicated mechanism of how this derived attribute, the net worth, is computed is hidden from the user.

Formally, the inheritance dimension of ownership is similar to that of Part-Whole which we have previously described. For further discussions of inheritance for part-of relationships, refer to our previous work [12].

7. IMPLEMENTATION

A preliminary implementation of a dimension-based ownership model is available on the web at
http://object.njit.edu:8080/ownership/ow_demo_op.html
with some introductory material at
http://object.njit.edu:8080/ownership/ow_demo.html.

The website contains the schema of an example from our domain. A person can own stock, bank accounts, patents, small business(es), building(s) and life insurance. A small business itself can own computer(s). The different kinds of owned objects require different kinds of ownership relationships. These different ownership relationships are characterized by different dimensions, according to our model. The current version of the program contains the basic functionality of creating, deleting, displaying, and changing objects and connecting them by ownership relationships. At this time, not all the dimensions have been implemented.

8. CONCLUSIONS

We have presented a methodology that distinguishes ownership from other semantic relationships in an object-oriented database, especially from Part-Whole, with six steps a database modeler must follow. Once a relationship is classified as one of ownership, it can be placed in a formal framework by defining seven characteristic dimensions: 1) exclusiveness, 2) dependency, 3) limitation, 4) status, 5) settlement, 6) documentation and 7) inheritance. The first six dimensions describe the ownership relationship and provide the necessary functionality to capture its complexities. The inheritance dimension is the same we presented in our work on the Part-Whole relationship. Just as in Part-Whole, inheritance provides the ability to combine identical properties from various sources, enabling declarative statements like "the value of John's assets is the sum of the valuations of all objects connected by ownership relationships to John."

Acknowledgment
This research was partially supported by the New Jersey Commission for Science and Technology through the New Jersey Center for Software Engineering.

References

1. Tony Baer, *Object databases*, Computerworld **33** (1999), no. 3, 66–67.
2. Francois Bancilhon, *Object databases.*, ACM Computing Surveys **28** (1996), no. 1, 137–140.
3. Elisa Bertino and Giovanna Guerrini, *Extending the ODMG object model with composite objects*, Proceedings of the Conference on Object-Oriented Programming Systems, Languages and Applications, 1998 **10** (1998), 259–270.
4. Elisa Bertino and Lorenzo Martino, *The object-oriented database systems: Concepts and architectures*, Addison-Wesley Publishing Company, 1993.
5. Conrad Bock and James Odell, *A more complete model of relations and their implementation. part i: Relations as object types*, Journal of Object-Oriented Programming **10** (1997), no. 3, Reprint: ©1997 SIG Publications, Inc, New York, NY.
6. ______, *A more complete model of relations and their implementation part ii: Mappings*, Journal of Object-Oriented Programming **10** (1997), no. 6, Reprint: ©1997 SIG Publications, Inc, New York, NY.
7. ______, *A more complete model of relations and their implementation part iv: Aggregation*, Journal of Object-Oriented Programming **11** (1998), no. 5, Reprint: copyright 1998 SIG Publications, Inc, New York, NY.

8. Securites Training Corportation, *Series 7: General securities nyse/nasd registered representative study guide*, Securities Training Corporation New York, New York, 1997.
9. M. Halper, J. Geller, and W. Klas, *Integrating a part relationship into an open oodb system using metaclasses*, Proceedings of the 3rd Int'l Conference on Information and Knowledge Management (Gaithersburg), 1994, pp. 10–17.
10. M. Halper, J. Geller, and Y. Perl, *An OODB "part" relationship model*, Proceedings of the First International Conference on Information and Knowledge Management (Baltimore, MD), 1992, pp. 602–611.
11. ———, *Value propagation in object-oriented database part hierarchies*, Proceedings of the 2nd Int'l Conference on Information and Knowledge Management (Washington, DC), 1993, pp. 606–614.
12. ———, *An OODB part-whole model: Semantics, notation and implementation.*, Data and Knowledge Engineering **27** (1998), 59–95.
13. Brian Henderson-Sellers, *Towards the formalization of relationships for object modelling*, Technology of Object-Oriented Languages and Systems, 1997 Proceedings TOOLS 25 (1997), 267–283, 24-28 Nov 1997.
14. Brian Henderson-Sellers and Franck Barbier, *What is this thing called aggregation?*, Technology of Object-Oriented Languages and Systems, 1999 Proceedings of (1999), 236–250.
15. M. Huhns and L. Stephens, *Plausible inferencing using extended composition*, International Joint Conference on Artificial Intelligence, IJCAI-89, 1989, pp. 1420–1425.
16. J.H.Reichman, *Electronic information tools-the outer edge of world intellectual property law.*, Dayton Law Review **17** (1992), 1.
17. L. Thorne McCarty, *Ownership: A case study in the representation of legal concepts (preliminary draft).*, Conference in Celebration of the 25th Anniversary of the Istituto Documentazione Giuridica (1993), 1–23.
18. ———, *An implementation of eisner v. macomber.*, International Conference on Artificial Intelligence and Law. Proceedings of the Fifth International Conference on Artificial Intelligence and Law (1995), 276–286.
19. ———, *Some arguments about legal arguments.*, International Conference on Artificial Intelligence and Law. Proceedings of the Sixth International Conference on Artificial Intelligence and Law (1997), 215–224.
20. Li min Liu and Michael Halper, *Incorporating semantic relationships into an object-oriented database system*, Systems Sciences, 1999. HICSS-32 Proceedings of The 32nd Annual Hawaii International Conference **32** (1999), 1–10, Held on 5-8 Jan 1999.
21. Robert N. Moles, *Logic programming-an assessment of its potential for artificial intelligence applications in law.*, Journal of Law and Information Sciences, 1991 **2** (1991), no. 2, 1.
22. James J. Odell, *Six different kinds of compositions*, Journal of Object-Oriented Programming **5** (1994), no. 8, Reprint: copyright 1994 SIG Publications, Inc, New York, NY.
23. W. Rahayu, E. Chang, and T.S. Dillion, *Implementation of object-oriented association relationships in relational databases.*, Database Engineering and Applications Symposium, 1998 Proceedings IDEAS'98 International (1998), 254–263.
24. Amir Sapir and Ehud Gudes, *Dynamic relationships and their propagation attributes in object oriented databases*, Database and Expert Systems Applications, 1998 Proceedings. Ninth International Workshop **9** (1998), 967–974, Held on 26-28 August 1998.
25. Lawrence B. Solum, *Essay:legal personhood for artificial intelligences.*, North Carolina Law Review **70** (1992), 1.
26. Wei Sun, Sha Guo, and F. Arafi, *Supporting inheritance in relationships in relational database systems.*, Software Engineering and Knowledge Engineering. 1992 Proceedings Fourth International Conference (1992), 511–518.
27. M.E. Winston, R. Chaffin, and D.J. Herrmann, *A taxomony of part-whole relations.*, Cognitive Science **11** (1987), no. 4, 417–444.

28. Oscar Yang, James Geller, Michael Halper, and Yehoshua Perl, *Modeling business applications with the OODB ownership relationship*, Third International Conference on Artificial Intelligence Applications on Wall Street (1994), 2–10.
29. Oscar Yang, Michael Halper, James Geller, and Yehoshua Perl, *The OODB ownership relationship.*, In Proceedings of the Int'l Conf. on Object Oriented Information Systems ed. D. Patel, Y. Sun and S. Patel, Springer-Verlag. (1994), 278–291.
30. Nan Zhang and Theo Harder, *On a buzzword "extensibility"-what we have learned from the orient project?*, Database Engineering and Applications, 1999 IDEAS'99 International Symposium Proceedings (1999), 360–369.