EXTENDING ENTITY/RELATIONSHIP FORMALISM FOR SPATIAL INFORMATION SYSTEMS

Dr. Yvan Bédard, Director
François Paquette, B.Sc.A., M.Sc. candidate
Laboratory for Spatial Information Systems
Geomatics Center
Dept. of Geodetic Sciences and Remote Sensing
Laval University
Sainte-Foy, Qc
Canada, G1K 7P4
(418) 656-3694
BITNET 1130025@LAVALVX1

ABSTRACT

Information Engineering develops and uses systematic methodologies and tools to facilitate the implementation of information systems. However, for Spatial Information Systems (SIS), these methodologies and tools need to be extended to better consider the spatial characteristics of the data and of their processing. One of these tools, the Entity/Relationship (E/R) formalism, is more and more used to build data models for SIS. This paper describes difficulties of the standard E/R formalism with regards to spatial phenomena and suggests three extensions to improve E/R effectiveness: the Sub-Model Substitution (SMS) technique, the inclusion of *cartographic only* objects, and generalization. This paper also provides additional research directions to improve E/R formalism expressive power.

INTRODUCTION

Over the last three years, we have seen more and more papers on the use of E/R modeling for Spatial Information Systems (SIS). Although this is a useful tool for the implementation of SIS, its expressive power is somewhat limited for spatial phenomena. Specialists in non-spatial information systems (e.g. banks, hospitals, schools) have been working with E/R for the last ten years and have recognized the need to improve E/R modeling for special purposes.

This paper begins with an overview of what data models are and what is their "raison d'être". Then come definitions and rules to build conceptual data models: the Individual Formalism, also called French E/R. Some problems related to the modeling of spatial data with such a formalism are presented. Then, it is used with three extensions to better model spatial phenomena: the Sub-Model Substitution (SMS) technique, the inclusion of *cartographic only* objects, and generalization. We close this paper with practical results and additional research issues.

SPATIAL DATA MODELING

Data models are simplified views of a part of the reality, they are built according to certain rules to facilitate the implementation of a database in an information system. Shlaer and Mellor (1988) mention that a data model is "a *thinking tool* used to aid in the formalization of knowledge". In fact, our general capability of understanding, remembering, making decisions and communicating depends upon our capability of making models.

Data modeling is an abstraction process where the essential elements are emphasized and the non-essential ones eliminated with regard to a specific goal (e.g. improve transportation, provide better management of property files). Data modeling requires the use of rules to create the model (e.g.Codd's normal forms) and to communicate this model, i.e. a language using a well-defined set of symbols (literal and/or graphical) with associated meanings.

Building good data models is very important since they play a major role in the determination of "which part" of the reality is being represented in the database, how it is represented, what can be done with this representation, and how fast it can be done. In addition, data models describe the most stable and expensive resource of an information system: data.

Creating data models is a multi-step mapping of the reality into a physical database and its representation to the users. Well-known examples are the three types of data models (internal, conceptual, and external schemas) identified by the ANSI/SPARC Study Group on Data Base Management Systems (1975). The **Conceptual Schema** is a representation of the reality showing all entity types to be included in the database, their attributes and their relationships. This view is independent of the type of Data Base Management Systems (DBMS) used. It is written in a simple language and is directed towards the system manager. From the programmers' point of view, this is a high level of abstraction of the database structure. At this level of abstraction, the expressive power of semantics formalisms such as E/R is necessary.

The **Internal Schema**, also called physical schema, shows the view of the reality as it is structured in the computer database, i.e. how data are physically stored and related. It usually is written with the Data Definition Language (DDL) of a DBMS or with standard programming languages such as PASCAL, FORTRAN or COBOL. Thus, the internal schema depends upon the software or language used for the implementation of the information system. This data model is written in a more complex language (programming code) adapted to programmers and computers. From the programmers' point of view, this is the lowest level of abstraction of the database structure. This data model can be a direct translation of the conceptual model or one optimized for better computing performance.

The **External Schema**, also called user schema, is an exact or modified subset of the conceptual schema. It is built to illustrate which entity types, attributes and relationships of the database are available for a specific use or user. It usually is written in the same simple language than the conceptual schema and is directed towards end users for specific applications. From the programmers' point of view, this is the highest level of abstraction of the database structure.

More and more, we use an additional data model to ease the translation from the conceptual schema to the internal schema: the **logical data model** (Bédard 1988). This logical schema depends on the type of DBMS used and is written in DBMS-oriented formalisms such as CODASYL and Relational.

Thus, to avoid the difficult task of going directly from "talking about" the reality we want to manage with the information system to "programming" its corresponding database, *different data models are needed at different levels of abstraction.* Such intermediary steps are especially useful when building large databases (as in most SIS). Sometimes, the term "datalogical" is used for the internal model and the term "infological" for the conceptual and external models; this indicates that the latter more closely represent the reality while the former rather represents the database.

Good data models

A good data model includes all the entity types that we want information about, all the attributes necessary to describe the desired characteristics of the selected entity types, and all the necessary relationships among these entity types.

In addition, a good data model eliminates redundancy. This can be done with Codd's (1972) Normal Forms which are well known by relational DBMS modelers. Redundancy elimination is usually done at the conceptual level, leaving the optimization of the data structure (to improve computing performance by reintroducing well chosen redundancies) for the internal model.

819

Data dictionary

To completely describe the reality (in a database sense), the data models must be completed by a data dictionary. Such dictionary contains all the necessary metadata about the data models. Usually, it includes the name and definition (including examples and exceptions if necessary) of the entity types and attributes included in the data model. It also includes the type (e.g. real, boolean) or each attribute as well as some integrity constraints (e.g. domain of values) and the measurement units used. The programmer should find all he needs to do his task in the data model and the data dictionary.

ENTITY/RELATIONSHIP FORMALISM AND ITS APPLICATION TO SPATIAL PHENOMENA

The objective of E/R modeling is "to create a description of the semantics of data that reflects the actual enterprise and its information requirements" (Martin and McClure 1988). These authors also add that "the task of the data modeler is to capture reality and communicate about it accurately. He tends to be distracted from this task if he has to think about computer hardware or data-base software or if the line between semantics and the implementation of data becomes blurred".

According to the E/R concept, we make conceptual data models by identifying, classifying, describing and relating parts of the real world to organize the information into a formal structure amenable to a computer form. Thus, it is useful to perceive the reality as containing "entities" or objects, "attributes" or characteristics of the objects, and "relationships" between entities:

Entity: object, person, concept or event about which we want information; the type of an entity is usually identified by a noun (e.g. entity type *River*, entity type *Road*).

Relationship: association between two types of entities; usually identified by a verb or a preposition (e.g. Road *to cross* River). A relationship has a **cardinality** giving the number of times (minimum and maximum) the relation can occur between two specific entities (occurences). For example, if we say that a Road crosses a River a minimum of 0 times and a practical maximum of 5; and on the other hand that a River can be crossed by a minimum of 0 Roads and an unknown maximum of N, this leads to a relation *to cross* with a cardinality of 0,5 in one direction and 0,N in the other direction.

Attribute: characteristic of an entity type or a relationship; for each entity, it contains a value called data; mostly identified by an adjective, a noun, or a group of nouns and/or adjectives (e.g. for Road: *pavement, number of lanes, number of accidents*). When an attribute is used to identify a specific entity (occurence) within its group of similar entities, this attribute is called **identifier** or key (e.g. for Road: name).

These three basic constructs of the Entity-Relationship model have been graphically represented several ways in the past few years, leading to different E/R representations of the same data model. Several examples can be found in the literature, for example Martin and McClure (1988) have identified three notation styles, each of them different from the original E/R style presented by Chen (1976) or from the styles used in the other references of this paper. For this paper, we use the French notation called "Individual Formalism".

According to this formalism, an entity type is represented by a rectangle containing its name at the top in uppercase letters. A relationship is represented by a line with a central ellipse containing its name at the top in uppercase letters. Attributes use

lowercase letters and are included either in the rectangle of the entity type they describe or in the ellipse of the relationship they describe. Attributes serving as entity identifiers are placed first in the list of attributes and are underlined. All entity names must be unique. The cardinality in one direction of the relationship is placed on the relation line, close to the entity from which we start reading the relationship. (for more information, see Collongues and al 1986, Tabourier 1986, or Tardieu and al 1986)
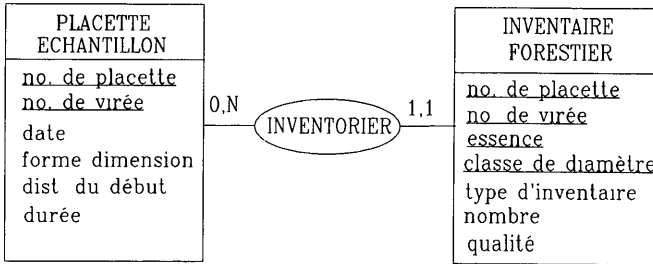
Figure 1: Individual Formalism diagram

This formalism, like other semantics formalisms, is built for a traditional database structure and is not well suited for spatial referencing (location and geometric description of entities) and SIS. For example, it does not easily allow the inclusion of primitive and complex geometric entities (e.g. points, edges, polygons, point sets, nodes, lines, networks, sets of polygons) with spatial attributes (e.g. coordinates, minimum bounding rectangle) and spatial relationships (e.g. connectivity), and the analysis of spatial operations (e.g. overlay, buffering, distance and area measurements, connectivity, intersection, spatial querrying). In addition, not all computerized cartographic objects can actually be represented.

Some problems and considerations

The actual way to deal with spatial referencing is either to avoid its modelization or to add it to the conceptual data model exactly like other entities, attributes and relationships. Avoiding the modelization of spatial referencing creates a conceptual data model not showing all the data available in the GIS database(s). This is a problem per se since the data model must show all entities, attributes and relationships to be included in the global database of an organisation (eitheir located in one or two different databases, graphic and non-graphic, in the GIS). We think that the data model should show which geometric entities are needed (e.g. points, lines, polygones, sets of points) to draw non-geometric entities (e.g. roads, rivers, houses). Knowing the needed geometric entities influences the choice of the GIS system to buy as well as the physical structure of the database. Also, the programmer needs to know all cartographic attributes to include in his program code (e.g. symbology, different geometric descriptions for different scales). All this information can be included either in the data model or the data dictionary.

On the other hand, including information about spatial referencing in a conceptual schema with the actual E/R formalism introduces other problems. The first one is related to the size of the conceptual schema. Very often, SIS are complex systems already involving a large number of entities and relationships and many more attributes, leading to data models complicated to draw, verify, modify, and read. The simple addition of geometric entities (with their attributes) and their relationships (1) among them, and (2) with the non-geometric entities, rapidly fills up the conceptual model, worsening the size problem. For example, every spatially referenced entity has a relationship with the geometric entity describing its shape. With such a solution, the information is available but slow to find because we must
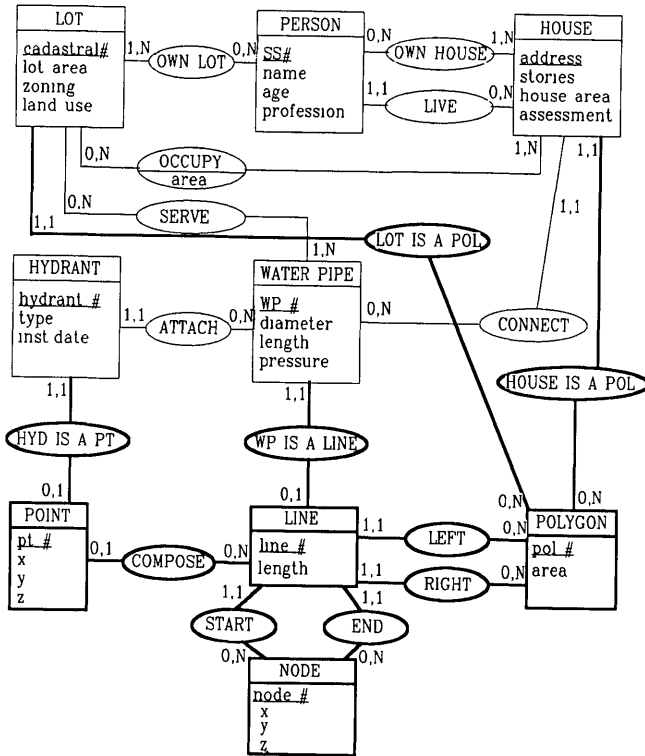
navigate in an already large conceptual model.

LOT
cadastral#
lot area
zoning
land use

1,N OWN LOT 0,N

PERSON
SS#
name
age
profession

0,N OWN HOUSE 1,N

HOUSE
address
stories
house area
assessment

1,1 LIVE 0,N

0,N OCCUPY
area

0,N SERVE
1,1

1,N

LOT IS A POL

HYDRANT
hydrant #
type
inst date

1,1 ATTACH 0,N

WATER PIPE
WP #
diameter
length
pressure

0,N

CONNECT

HOUSE IS A POL

1,1

HYD IS A PT

1,1

WP IS A LINE

0,1

POINT
pt #
x
y
z

0,1 COMPOSE 0,N

0,1

LINE
line #
length

1,1 LEFT
1,1 RIGHT

0,N POLYGON
pol #
area

1,1 START END 1,1

0,N NODE
node #
x
y
z

Figure 2: explicitly including (in bold) geometric entities and relationships (from Bédard 1988)

In addition, including the geometric entities in the conceptual model raises the question of "which geometric data structure to use?". This simple question brings up a whole new problem that GIS researchers have worked on for years (e.g. spaghetti, CAD/CAM type structures, topology, Voronoï Diagrams, Cell graphs). We think that this is a problem for GIS developpers and scientists and that it should be kept away from GIS users while modeling their database. Furthermore, since one difference between DBMS and GIS systems is that DBMS have "no predefined data structures" while GIS provide "built-in data structures" for geometry, why go into the details of modeling a geometric data structure?

It also is interesting to note that for most GIS users, geometric entities are not real life objects and have no meaningful attributes. Since they are perceived differently than other entities, why treating points, lines and polygons as we treat buildings, roads, and forest cells in the data model? On the other hand, their geometric description is as important as their non-graphic attributes. So, why not indicate the geometric shape of an object with the other attributes, i.e. in the data model?

There is an additional problem with traditional E/R modeling: only the entity types explicitely defined in the database are represented, i.e. only the objects having their own object file containing attributes. This represents a limitation for SIS objects appearing on a map but not needing an object file in the database. These objects appear on maps either as background information or as objects to manage based only on their geometric properties, cartographic layer or symbolic value (e.g. road

types on topographic maps). At first sight, these objects may not be perceived as entity types, but if we consider the conceptual data model as a thinking tool (1) to define what information is available and (2) as a necessary step helping the programmers' task, and if we remember what an entity is, then these objects can be treated as entities.

Furthermore, we are investigating if the conceptual data model should describe *all* the information of an organisation and not only the information to be computerized. Actually, it seems reasonable to think so, especially for SIS which are large systems where not all cartographic information is necessarily computerized. We would then need further extension to E/R.

EXTENDING E/R FORMALISM

Data modeling is still evolving and specific improvements are needed in specific fields, leading to specific data modeling techniques. In a recent research project, we found that the major weakness of actual information system design methlogies, when applied to SIS, is their modelization tools (Boutin 1988). For SIS applications, we need an extended formalism allowing us to include semantics specific to our purposes and to better represent all entities, how they are located, geometrically defined, and spatially interrelated. Such semantics must be included either in the data model or its corresponding data dictionary. Brodie (1984) mentioned that a new generation of data models is emerging: special purpose data models (for applications such as VLSI, CAD/CAM and Cartography). The ideas presented hereafter (SMS technique, including *cartographic only* objects, and generalisation) are still in development but represent a step in the building of such a new generation of formalisms for spatial data modeling.

The Sub-Model Substitution (SMS) technique

As previously mentioned, one of the actual problems with basic semantic modeling such as E/R is that their models rapidly become complex for large databases, making it easier to leave redundancies, inconsistencies and omissions out. Most of this can be solved by a good CASE program (Computer-Assisted Systems Engineering) providing data modeling tools; however, the difficulty involved in reading large models remains and this only is with more powerful modeling rules that readibility will improve.

The SMS technique uses a set of meaningful graphical symbols to replace the relationships between the non-geometric entity types of the database (e.g. roads, rivers, forest cells) and the geometric entity types describing their cartographic shape (e.g. lines, networks, polygons). The SMS technique is built to encourage simplicity of building, verifying, modifying and reading.

SMS Rule: if an entity type has a geometric description, the original geometric description is represented by a graphical symbol placed next to the name of the entity type, on its left hand side. Each graphical symbols of figure 3 is a substitution of its corresponding sub-model.

The list of geometric descriptions presented in figure 3 still is in development, however we can already see advantages to the SMS technique: (1) to almost eliminate one relationship per entity type, and (2) to eliminate the need to create a geometric data structure, thus (3) to eliminate all geometric entity types with their attributes and interrelationships. In a data model of medium complexity (46 entity types, 95 relationships, 167 attributes) for a forest application, this has resulted in a reduction of 47 relationships plus a few entity types and attributes (Paquette 1988). The simplicity and usefulness of the SMS technique has already been accepted by two major SIS consulting companies in Québec who have begun to use the idea.

SMS Symbol	Geometry	Sub-model substituted by symbol
•	point	1,1 —— DESCRIBE —— 0,1 POINT
⌒	line	1,1 —— DESCRIBE —— 0,1 LINE
⋁⌒	polyline	1,1 —— DESCRIBE —— 0,1 POLYLINE
⋎⌐	simple network	1,1 —— DESCRIBE —— 0,1 SIMPLE NETWORK
⬠	simple polygon	1,1 —— DESCRIBE —— 0,1 SIMPLE POLYGON
◁	complex network	1,1 —— DESCRIBE —— 0,1 COMPLEX NETWORK
ᗣᗤ	joint polygons	1,1 —— DESCRIBE —— 0,1 JOINT POLYGON
⊞	partition	1,1 —— DESCRIBE —— 0,1 PARTITION

Figure 3: SMS symbols with their substituted sub-models.

Including *cartographic only* objects

As previously mentioned, traditional E/R modeling does not accept objects appearing on a map but not needing an object file in the database. This can be solved by the simple rule presented in the next paragraph and which completes very well the SMS rule both in content and readibility.

Cartographic Only Objects Rule: if an entity type has non-geometric attributes (e.g. Road: pavement, number of lanes, number of accidents) explicitly stored in the database , i.e. not deduced from the symbology of the base map, then a *database* or *hard disk* symbol must be placed next to the name of the entity type, on its right hand side.

▣ PLACETTE ⊟ ECHANTILLON	▥ COURS D'EAU	INVENTAIRE ⊟ FORESTIER
no. de placette	nom du cours d'eau	no. de placette
no. de virée	largeur	no. de virée
date	type	essence
forme dimension		classe de diamètre
dist. du début		type d'inventaire
durée		nombre
		qualité
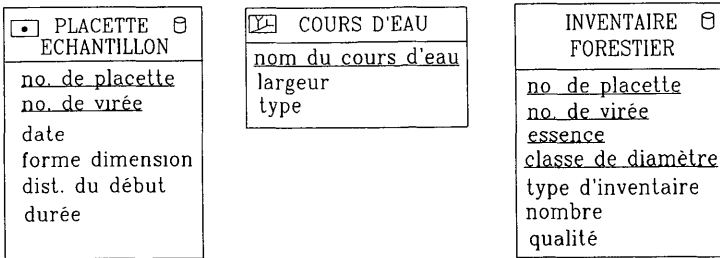
Figure 4: inclusion of the database symbol: (1) a mapped entity type with DBMS attribute file, (2) a mapped-only entity type, and (3) a non-mapped entity type.

This rule allows us to (1) to add in our conceptual data model those useful objects which are on a map but which need no database files per se, (2) to clearly differentiate between entity types which are *mapped only* and those which are *mapped and stored in the database as entities*, (3) to know this before building the next data models (logical and internal) which need this information, and (4) to help in the definition of background information for map presentation.

Generalisation

Generalisation allows us to create meaningful groups of entities with common characteristics (e.g. forest operations: spraying, cutting, soil preparation, thinning out, planting, etc.). This is not a new idea in semantics formalism, and it has already

been introduced in the field of SIS (cf. Frank 1985, Blais 1987, Egenhofer 1987). However the following paragraphs state the necessary rules to integrate generalisation with the previous two techniques:

Generalisation Rule 1: a super-entity type can be created from a logical group of entity types all sharing one or more common attributes with common domains of possible values; each common attribute becomes an attribute of the super-entity type. This rule indirectly states that (1) two attributes with different domains of possible values are considered different and must have two different names, and (2) all sub-entity types have, in addition to their own attributes, the super-attributes;

Rule 1.1: for the identifiers, there is the additional condition of unicity among generalized entities; often this may lead to extend the domain of values (e.g. sequential numbers).

Generalisation Rule 2: a super-entity type can only have relationships common to all sub-entity types; when such a relationship exists, all sub-entity types are logically (but not graphically) connected to this relationship. When such relationship does not exist, the relationships directly go to the sub-entity types. This rule indirectly states that a sub-entity type has all the relationships of its super-entity type.

Generalisation Rule 3: the geometric description of a super-entity type is deduced from the geometry of its sub-entity types. For example a super-entity type, such as an hydrographic network, grouping linear (rivers) and polygonal (lakes) elements can be a complex network but not a point. Similarly, a super-entity type grouping only polygonal elements can be a polygon but not a line.

Rule 3.1: if all sub-entity types are mapped and have identical geometrical descriptions, then the super-entity type inherits this geometric description and becomes the only one needing to show the SMS symbol (all sub-entity types sharing this symbol with the super-entity type). For example, the Forest Operation entity-type would show the polygonal symbol, but the sub-entities Spraying, Cutting, Soil preparation, Thinning out, and Planting would show no SMS symbol. On the other hand, if the sub-entity types have different geometrical descriptions, they all keep their SMS symbol and the super-entity type shows its own (e.g. Hydrographic Network, River and Lake entity types all show their own SMS symbol).

Figure 5: example of generalisation.

Generalisation Rule 4: a super-entity type inherits the database symbol only if all its sub-entity types have one; then, the sub-entity types do not need to show this symbol anymore (because they now share it with the super-entity type). If not all sub-entity types have the database symbol, the ones having this symbol keep it and the super-entity type shows no such symbol.

Generalisation Rule 5: super-entity types can also be generalised.

Applying these rules to the previously mentioned forest example has led to a reduction of 74 attributes and 36 relations for an addition of 5 super-entities.

CONCLUSION

A good formalism at the *conceptual level* must be simple to understand, have a strong expressive power, and be as rigourous as possible. The development of the three extensions presented in this paper aims to meet this goal. For example, the modelisation of a medium size database for a forestry application has allowed us to include *cartographic only* objects in the conceptual data model and to reduce its size by 50% while improving its readibility.

Such results have shown to be useful and practical. However, this still is in development and additional research issues remain for SIS data modeling: modeling spatial relationships, generalizing relationships, dealing with multiple geometric descriptions of entities (e.g. at different scales), building rigourous rules to translate the conceptual schema into the logical and internal schemas, extending the SMS technique to 3-D, extending the data dictionary, applying the SMS technique to management of time-related entity types and relationships, and dealing with mutual exclusivity and mutual inclusivity of relationships.

REFERENCES

ANSI/X3/SPARC Study Group on Data Base Management Systems 1975, Interim Report, FDT (ACM SIGMOD Bulletin) 7, No. 2.

Bédard Y. 1988, On Spatial Data Modeling (First Draft). Third Internal Seminar on Trends and Concerns of Spatial Sciences, Dept. of Geodetic Sciences and Remote Sensing, Laval University, Québec City, June 6-8.

Blais R. 1987, Theoretical Considerations for Land Information Systems: <u>Canadian Surveyor</u>, Vol.41, No.1, pp. 51-64.

Boutin G. 1988, <u>Etude de l'applicabilité d'une méthode traditionnelle de conception de système d'information dans le contexte d'un système d'information à référence spatiale</u>: M.Sc. Thesis, Dept. of Geodetic Sciences and Remote Sensing, Laval University, Québec City, 206 p.

Brodie, M.L. 1984, On the Development of Data Models: in Brodie M.L., J. Mylopoulos and J.W. Schmidt, <u>On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases, and Programming Languages</u>: Springer-Verlag, New-York, pp. 19-47.

Chen P. 1976, The Entity-Relationship model: Toward a Unified View of Data: <u>ACM Transactions on Databases Systems</u>, 1,1, pp. 3-36.

Codd E.F. 1972, Further Normalization of the Data Base Relational Model: Prentice-Hall, <u>Data Base Systems</u>, Courant Computer Science Symposia Series, V.6.

Collongues A., J. Hugues and B. Larouche 1986, <u>MERISE méthode de conception</u>: Dunod Informatique Ed., 211 p.

Date C.J. 1986, <u>An Introduction to Database Systems, V.1, 4th Ed.</u>: Addison-Wesley, 639 p.

Egenhofer M. 1987, Appropriate Conceptual Database Schema Designs for Two-Dimensional Spatial Data Structures: Technical Papers, ASPRS-ACSM Annual Convention, V.5, Baltimore, March 29-April 3, pp. 167-179.

Frank A. 1985, Class notes: Surveying Eng. Dept., Univ. of Maine, Orono, USA.

Martin J. and C. McClure 1988, <u>Structured Techniques: the Basis for CASE, Revised Edition</u>: Prentice-Hall, 776 p.

Paquette F. 1988, Utilisation et adaptation du formalisme entité-relation pour la structuration d'une base de données à référence spatiale avec application à la gestion des données forestières. Unpublished paper presented at the Dept. of Geodetic Sciences and Remote Sensing, Laval University, Québec City, Dec. 7.

Shlaer S. and S.J. Mellor 1988, <u>Object-Oriented Systems Analysis: Modeling the World in Data</u>: Yourdon Press (Prentice-Hall), 144 p.

Tabourier Y. 1986, <u>De l'autre côté de MERISE</u>: Editions d'Organisation, 241 p.

Tardieu H., A. Rochfeld and R. Colletti 1986, <u>Méthode MERISE, principes et outils, Tome 1, 2nd ed.</u>: Editions d'Organisation.

Ullman J.D. 1982, <u>Principles of Database Systems</u>: Computer Science Press, 484 p.

Wang F. and Newkirk R. 1988, An Entity-Relationship Model for Geographical Information System Development: ACSM-ASPRS Annual Symposium, Saint-Louis, V.5, p. 162-172.