

# **A PRESENTATION LANGUAGE FOR GIS CADASTRAL DATA**

GILBERTO CÂMARA

National Institute for Space Research (INPE), Brazil (gilberto@dpi.inpe.br)

MARCO ANTONIO CASANOVA

Latin American Center for Higher Education and Research, IBM (casanova@vnet.ibm.com)

UBIRAJARA MOURA DE FREITAS

National Institute for Space Research (INPE), Brazil (bira@dpi.inpe.br)

JOÃO PEDRO CERVEIRA CORDEIRO

National Institute for Space Research (INPE), Brazil (jpedro@dpi.inpe.br)

LAURO HARA

National Institute for Space Research (INPE), Brazil (lauro@dpi.inpe.br)

## **ABSTRACT**

This paper introduces a presentation language for GIS. This language is designed to be used a standalone tool, or in conjunction with a query and manipulation functions. The proposed language is based on an object-oriented data model and on concepts used in the object-oriented query language proposed standard OQL (Object Query Language). The language is part of the SPRING environment, a GIS system which integrates the diverse types of spatial data (cadastral maps, thematic maps, digital terrain models and images).

## **1. INTRODUCTION**

The widespread availability of low-cost presentation systems for GIS data has led to a unprecedented popularity of the technology. These packages are often termed "maptop systems", since they are most naturally geared towards an environment where the user has acquired the data elsewhere or has prepared it with the use of more complete GIS systems.

These systems present a number of similarities in their functionality. The most common task performed is the selection of a subset of the available data (usually a specific attribute of a relation containing geographical objects) and its presentation in the form of a choropleth map. These tasks are controlled by means of a menu-driven graphical user interface.

Permission to make digital/hard copies of all or part of this material for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copyright is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires specific permission and/or fee

GIS 96 Rockville MD USA

Copyright 1997 ACM 0-89791-874-6/96/11 ..\$3.50

In many applications, however, this menu-driven procedures are not sufficient, and some means of preparing customised scripts is necessary. This will be the case in municipal applications where questions about land ownership, real-estate values and associated taxes are to be asked repeatedly.

Recognizing such needs, some desktop GIS are now offering programming tools (such as ARC/VIEW's AVENUE) which provide the means for writing dedicated applications. Such approach provides the user with a large flexibility, but requires a significant effort of learning and programming. As a compromise between the flexibility and work of a programming environment and the simple but fixed nature of menu-driven systems, we propose the adoption of a language for GIS data presentation.

This language is a formal basis for the development of customized applications, graphical user interfaces or direct interaction using the commands and macro constructs.. This presentation language is to be used in conjunction with a query and manipulation language for GIS data, called LEGAL.

This work has also had a practical motivation. The language is part of the development of SPRING, a geographical information system designed to support environmental projects over large spatial data bases.

SPRING and is being built as part of the mission of INPE to develop advanced GIS and remote sensing software to help manage the country's vast environmental resources, such as the Amazonia Rain Forest. For further information on SPRING, please see [CSF+96].

This paper is organised as follows. In section 2, we analyse the requirements for a presentation language for GIS. In section 3, we review the proposals for GIS query and presentation languages found in the literature. The description of the proposed presentation language is given in section 4. Finally, the use of this presentation language within the SPRING environment is presented in section 5.

## 2. REQUIREMENTS FOR PRESENTATION LANGUAGES

A discussion on the specific requirements for presentation languages for GIS is found in [Ege90, Ege94]. Egenhofer suggests that a spatial query should consist of three parts:

- a) the description of the set of objects to be retrieved.
- b) a set of display queries, which separate query results into more detailed sets, each to be displayed in an individual format.
- c) a display description specifying how to render the data.

Egenhofer also argues that, while the requirement (a) might be fulfilled by a query language which is an extension of existing environments such as SQL, requirements (b) and (c) need a separate *presentation language*, which should cater for [Ege94]:

- *Assignment of graphical and pictorial descriptors to the geographical objects*: association of legends and visual definitions (colour, symbols) to the geographical objects.
- *Presentation control*: use of different graphical descriptors (colour, text, legend, scale) to differentiate the query results.
- *Combination of query results*: the user should be able to start a new picture, or to combine the result with the current display, by means of operations such as union, intersection and difference.
- *Presentation of spatial context*: the answer to a spatial query usually needs a context for its interpretation. For example, a query showing all hospitals which are located in a county also needs to present the county boundaries, as well as road information.

- *Additional information*: presentation of associated information by means of pictorial means such as cartograms, bar graphs and pizza diagrams.

These requirements need also to be completed with general principles for good graphical design. In this context, Bertin's system of "visual variables" [Ber73] is a general guideline for displaying quantitative and qualitative information. Bertin considers six "visual variables" (shape, orientation, colour, texture, value and size) and argues that our perception is enhanced when we use a continuous gradation of one of these variables, and that we become confused when we try to interpret graphics where these variables are mixed with consistency.

## 3. PROPOSALS FOR GIS QUERY AND PRESENTATION LANGUAGES

Earlier work on query languages for spatial data has focused on the aspects of spatial operations and on extensions of SQL which include spatial operations. These ideas, in turn, have led to extensive research on the nature of spatial relationships. For example, formalisms for topological relationships include the *4-intersection* [Ege90] and the *9-intersection* [EDH94]. In [Ege94] *SpatialSQL*, a spatial query and presentation language, was introduced. Other proposals for query languages based on SQL can be found in [Goh89, Ooi90].

An analysis of the issues regarding the use of SQL extensions for GIS query and presentation is given in [Ege92], which concludes that there are two major deficiencies of such languages: (a) the difficulty of incorporating graphical display and specification into SQL; (b) the lack of support with the relational framework for aspects such as knowledge and metadata queries.

Due to the shortcomings and problems of SQL, a number of researchers have attempted to use the framework provided by object-oriented data base management systems, especially the definition of an Object Query Language (OQL), a proposal for an object query language put forward by ODMG [Cat93]. Extensions of OQL to provide for spatial data queries are examined in [SVP+96].

One of the important differences of the OQL proposal (as regards the traditional SQL approach) is the availability of *collections*, which are ordered sets of objects of the same type. In this paper, we have examined the idea of using *collections* (which store query results) as a basis for defining a presentation language.

Using object-oriented query languages instead of relational ones for geographical data allows for added flexibility in dealing with issues such as: definition and manipulation of query results, multiple representations

assigned to the same object and assignment of presentation control to a collection of geo-objects.

#### 4. DESCRIPTION OF QUERIES AND PRESENTATIONS IN LEGAL

##### 4.1 GENERAL DESCRIPTION

The presentation language described in the paper, is part of a more general language for GIS, called LEGAL. The following discussion includes the general concepts that have influenced the design of the presentation module.

LEGAL is a spatial query and manipulation language which is based on a general data model for GIS data, which caters for the concepts of fields and objects [CFS+94], and which includes *local*, *focal* and *zonal* operations over geo-fields, *spatial selection* and *spatial join* operations over geo-objects, transformations from geo-objects to geo-fields (such as *attribute reclassification* and *buffer maps*) and from geo-fields to geo-objects (*spatial interpolation and identification*).

LEGAL is *strongly typed*. In LEGAL, *spatial queries* over geo-objects are implemented using OQL ("Object Query Language") extensions, and *manipulation operations* over geo-fields and *transformations* between geo-fields and geo-objects are implemented by operators at the same semantic level as OQL statements. For a complete description of LEGAL, please see [CCF96].

The following section describe the underlying data model used for cadastral GIS data, and the presentation language syntax.

##### 4.2 DATA MODEL

LEGAL deals with the two basic classes of objects: GEO-FIELD and GEO-OBJECT. The instances of GEO-FIELD, called *geo-fields*, describe continuous geographical variables over some region of the Earth, and have the following attributes:

- *location*: describes a geo-region  $R$ ;
- *range*: describes a set of values  $V$ ;
- *mapping*: describes a function  $\lambda: R \rightarrow V$ , which models the relation between locations in  $R$  and values in  $V$ .

The instances of GEO-OBJECT, called *geo-objects*, represent individualizable entities of the geographic realm. They are phenomena that may have one or more *graphical representations*, which correspond to the geo-referenced set of co-ordinates that describe the object's location.

Therefore, given a set of geographical regions  $R_1, \dots, R_n$ , a geo-object has the following in-built attribute:

- *location*: the set of regions  $r_1, \dots, r_n$ , where  $r_i \in R_i$  indicates the geographical location associated to the geo-object in the geographical region  $R_i$ .

Since most applications do not deal with isolated elements in space, it is convenient to store the graphical representation of geo-objects together with its neighbours. These features lead us to introduce the concept of *geo-object maps*, which group together the representations of geo-objects for a given cartographic projection and geographical region. Therefore, the representations for *geo-objects* are maintained in instances of the class OBJECT-MAP.

This view implies that *maps* are not modelled as *geo-objects*, but rather as geometrical representations which store the geographical locations of geo-objects.

This definition allows for multi-scale, multi-tile and multi-temporal representations to be associated to the same geo-object. To illustrate the concept, consider figure 1, which illustrates a data base for rivers of the Brazilian Amazonia. Since the region covers a very large area, a geographical data base in the 1:250000 scale (on UTM projections) will consist of several non-overlapping tiles. We associate each UTM partition to an instance of the class OBJECT-MAP which includes a mapping for all rivers which are included in the geographical area of the partition. Therefore, the Amazon river is seen by the data base as a single object, even though its representation may span several maps.

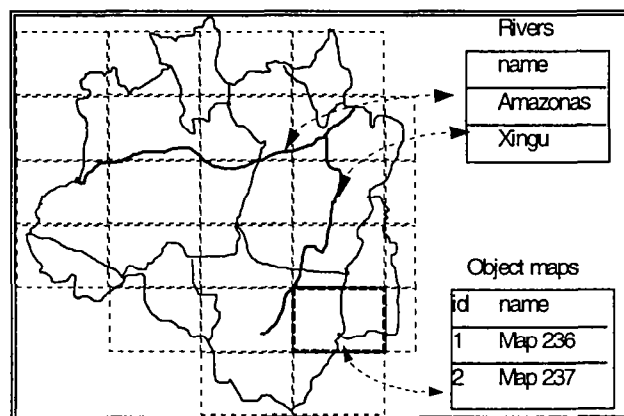


Figure 1 - Geo-Objects and Object Maps.

##### 4.3 QUERIES IN LEGAL

Queries in LEGAL have two components: a *search expression* expressed in an extension of OQL and a *query result* which may be used for further manipulation. For geo-objects, LEGAL offers spatial queries, computed based on

a geometrical representation. The spatial queries allow the topological restrictions INSIDE, TOUCH, CROSS, OVERLAP and DISJOINT, as defined by [CFO93].

The data model used in LEGAL allows a geo-object to be stored in more than one representation (which are instances of the class OBJECT MAPS). Therefore, in the case of multiple representations associated to one geo-object, the language allows for the possibility of defining which particular representation is used for computing spatial relationships. This option is indicated by the clause REPRESENTED BY.

For example, the spatial query  $Q_1$  = "Select the names of all native reservation areas which are located at less than 50 km from the Amazon river, in the geographical area surrounding the city of Manaus", is expressed in LEGAL as:

```
SELECT reserv.name
FROM river IN rivers_Brazil,
     reserv IN reserv_Brasil
WHERE river.name = "Amazon"
AND distance (reserv, river) < 50 km
REPRESENTED BY
    (SELECT map FROM map in Maps-Brasil
     WHERE map.name = "Manaus")
```

The construct REPRESENTED BY indicates that the locations of geo-objects in the classes Reservations and Rivers should be obtained from the object map named Manaus, retrieved by the nested query. This construct affects the semantics of the spatial function distance. This construct can be omitted, if there is only one representation associated to the geo-objects involved in the spatial query.

#### 4.4 MANIPULATION OF COLLECTIONS IN LEGAL

LEGAL offers specific constructs to enable the manipulation of query results, based on the ODMG model [Cat93]. The ODMG model enables the definition of *collections* containing an arbitrary number of objects of the same type. Therefore, the result of any selection, performed by LEGAL, can be assigned to an instance of the class COLLECTION.

To illustrate the use of collections in LEGAL, consider query  $Q_2$  = "select all hospitals located in the borough of Green Village", now expressed as a program which retrieves a collection of geo-objects.

```
COLLECTION hosp_gv (Hospital);
hosp_gv = (SELECT hosp
```

```
FROM hosp IN Hospital
     borough IN Borough
WHERE borough.name = "Green Village"
     AND hosp INSIDE borough);
```

#### 4.5 PRESENTATION LANGUAGE SYNTAX

Based on the requirements and principles expressed on section 2, and on the data model and the idea of *collections* presented above, the presentation language associated to LEGAL uses a syntax with three main components:

- the DEFINE VISUAL statement, which assigns graphical and pictorial descriptors to the geographical objects, such as legends, colour, texture and symbols.
- the SET MODE statement, which allows for combination of query results.
- the SHOW-AS-GROUP BY-WHERE statement, which caters for presentation control, including the possibility of generating additional information, such as bar charts.

##### 4.5.1 Visual Definition of Geographical Objects

The language allows for the definition for visual definitions associated to either *classes* of geo-objects, *individual* geo-objects or to *collections* of geo-objects. The visual characteristics include symbols, text, colour, line style and fill area pattern. The general definition for the visual definition for classes is expressed as:

```
DEFINE VISUAL FOR CLASS <object class>
    COLOUR <color>
    PATTERN<NONE | SOLID | DASH | HATCH >
    HATCH <hatch angle>
    SYMBOL <symbol name>
END
```

##### 4.5.2 Combination of Query Results

In a similar fashion to the Spatial SQL language [Ege94], the SET MODE statement controls the combination of query results, with the following syntax:

```
SET MODE < NEW | UNION | INTERSECT |
REMOVE >
```

where:

- NEW indicates that the display area should be cleaned before the results of the current query are shown ;
- UNION combines the results of the current query to the existing picture;
- INTERSECT selects all the objects which are both on the display and in the current query result
- REMOVE erases the result of the current query from the existing picture.

### 4.5.3 Presentation Control

The SHOW-AS-GROUP BY-WHERE statement controls the visualisation of the selected geo-objects or values (expressed as a collection containing query results) with the following syntax:

```
SHOW <collections | values >
AS < SHADING | SYMBOL | CARTOGRAM |
    CHART | TEXT >
GROUP BY < VALUES | INTERVALS >
WHERE <parameters>
```

The AS clause controls the graphical and pictorial description, which the following options :

- SHADING: the selected geo-objects are shown as filled-area polygons.
- SYMBOL: associates symbols to the selected geo-objects.
- CARTOGRAM: presents the values of a chosen attribute of the selected geo-objects as circles whose area is proportional to the value of the attribute. Its behaviour is affected by GROUP BY clause.
- CHART: presents the values of a chosen attribute of the selected geo-objects in a graphical form (bar charts, pizza diagram, dispersion plots). Its behaviour is affected by GROUP BY clause.
- TEXT: associates a descriptive text to the selected geo-objects.

The GROUP BY clause indicates the type of grouping to be performed in the geo-objects for presentation, with two options:

- GROUP BY VALUES : the geo-objects are grouped by values (such as quartiles or quintiles).
- GROUP BY INTERVALS: the objects are grouped, based on given intervals.

The WHERE clause indicates the associated parameters to the each presentation type.

In the case where GROUP BY VALUES has been selected, the geo-objects can be grouped in two, three, four or five groups ( HALFS, THIRDS, QUARTILES e QUINTILES), defined by the parameter:

```
VALUE_MODE= HALFS | THIRDS | QUARTILES
            | QUINTILES;
```

In the case where GROUP BY INTERVALS has been selected, the characteristics of the intervals need to be defined. The default behaviour is to have equally-spaced intervals, with maximum or minimum values, but is also possible to define unequally-space intervals, which the following parameters:

```
INTERVAL_MODE = EQUAL | UNEQUAL
INTERVAL_MAX  = <maximum value>
INTERVAL_MIN  = <minimum value>
INTERVAL_COUNT= <number of intervals>
INTERVAL_1=[<min_value>, <max_value>]
INTERVAL_n=[<min_value>, <max_value>]
```

The SCALE parameter affects the GROUP BY clause, and enables scaling the chosen attribute values of the collections. The use of non-linear scaling is important to avoid distortions in result presentations, as discussed in [Mon92]. Two options are available, as follows:

```
SCALE = LINEAR | LOGARITHM;
```

The presentation of choropleth maps (specified by the AS SHADING clause), is controlled by the following parameters:

- SHADING\_MODE: controls the choice between the options: shaded polygons of a given colour (COLOUR), a continuous colour palette (PALLETE), a single pattern (PATTERN), or a sequence of hatched-filled areas (HATCH).

```
SHADING_MODE = < NONE | COLOUR |
                PALLETE | PATTERN | HATCH >
```

- SHADING\_COLOUR: controls the shading colour, and is valid when SHADING\_MODE = COLOUR.

```
SHADING_COLOUR = <colour>
```

- SHADING\_PALLETE\_MIN and SHADING\_PALLETE\_MAX: controls the colours associated to a continuous palette (considered as a hue circle which ranges from purple to red, in a sequence which includes blue, cyan, green, yellow and orange). The initial and final colours are established by:

```
SHADING_PALLETE_MIN = <colour>
```

```
SHADING_PALLETE_MAX = <colour>
```

- SHADING\_PATTERN: controls the use of patterns for presentation. Patterns are indicated by a name.

SHADING\_PATTERN = <pattern\_name>

- SHADING\_HATCH: controls the use of hatched patterns for presentation, which two options, the angle and whether single or doubly-hatched lines are used.

SHADING\_HATCH\_ANGLE = <angle>

SHADING\_HATCH\_MODE=<SINGLE|DOUBLE >

The use of symbols (indicated by the AS SYMBOL clause), is controlled by the following parameters:

- SYMBOL\_NAME: indicates the symbol name.
- SYMBOL\_COLOUR indicates the symbol colour.
- SYMBOL\_SIZE determines the symbol size.
- SYMBOL\_ANGLE : indicates the symbol angle.

The use of cartogram (indicated by the AS CARTOGRAM clause) is affected by the GROUP BY clause (with the associated value and interval control) and by the parameter CARTOGRAM\_COLOUR, which indicates the colour of the circles.

The presentation of GIS results as charts (controlled AS CHART clause) with the options:

- CHART\_SHAPE, which controls the graph type :

CHART\_SHAPE = < 2DAREA | 3D AREA |  
2D\_COLUMNS | 3D\_COLUMNS | 2D\_BAR |  
DISPERSION | PIE\_CHART >

- CHART\_COLOUR, which controls the colours associated to the displayed values:

CHART\_COLOUR\_1 = <colour of the first variable>

CHART\_COLOUR\_n = <colour of the n-th variable>

The presentation of text ( indicated by the AS TEXT clause) can be controlled by the following parameters:

- TEXT\_NAME: indicates the text name.
- TEXT\_COLOUR: indicates the text colour.
- TEXT\_SIZE determines the text size.
- TEXT\_ANGLE indicated the text angle.

#### 4.5.4 PRESENTATION LANGUAGE EXAMPLE

To illustrate the presentation language, we shall consider the case of a study on Brasil's regional diversity on wealth. We shall use data on the distribution of poverty as of year 1985, as provided by the Brazilian Census

Bureau, where "poor" families are so consider those where the household income is less than Us\$ 50 dollars a month. This data are shown in Table 1.

TABLE 1  
PERSONS WITH LESS THAN 50 USD/MONTH  
HOUSEHOLD INCOME

<i>Region</i>	<i>Number (thousands)</i>
North	1.325
NorthEast	25.830
SouthEast	16.076
South	6.551
Center-West	3.389

Source: Brazilian Census Bureau (1985).

Initially, we would like to display the map of Brasil and the poverty maps by means of a cartogram, as shown in the following program.

```

COLLECTION map_reg (Map);
COLLECTION regions (Regions);
SET MODE new;
map_reg = (SELECT map FROM map IN Map
WHERE map.name =
"mapa_regioes_BR");
SHOW map_reg
AS SHADING
WITH SHADING_MODE = NONE;
SET MODE overlay;
regions= (SELECT reg
FROM reg IN Regions);
SHOW regions.poverty
AS CARTOGRAM
WHERE CARTOGRAM_COLOUR = "GRAY"
AND SCALE = LINEAR;

```

The result is shown in Figure 2.

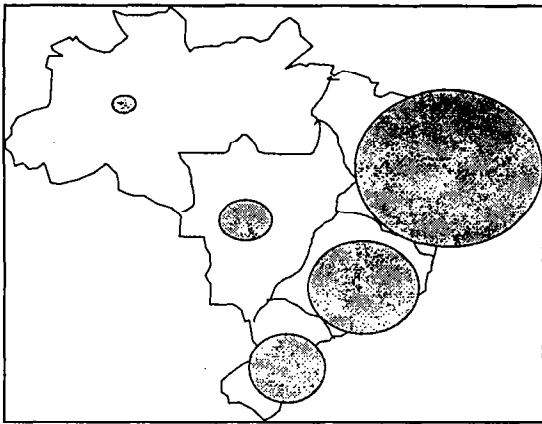


Figure 2 - Result of combination of query results.

We may also be interested in showing this data by means of a bar chart:

```

SET MODE NEW;
SHOW regions.poverty
  AS CHART
  WHERE CHART_MODE = 3D_COLUMN
  AND SCALE=LINEAR;

```

The result is shown in Figure 3.

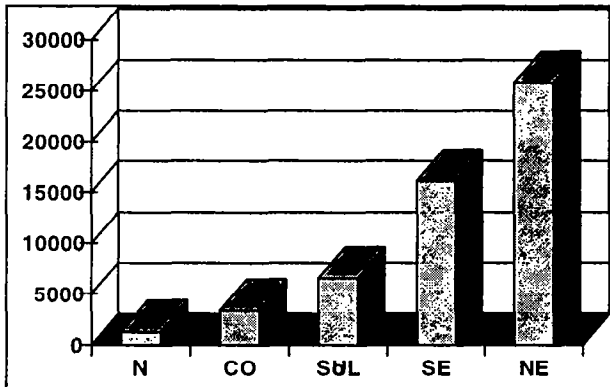


Figure 3 - Poverty distribution in Brazil.

## 5. CONCLUSIONS

The authors have discussed a query and presentation language for spatial data which is based on an object-oriented data model and uses concepts that have been proposed for object-oriented query languages such as OQL. The proposed language is based on the idea of using *collections* as sets of objects retrieved from a spatial query. These collections of objects will be displayed according to a specific set of control parameters. The proposed language is general enough to cater for the vast majority of

applications of GIS cadastral data presentation, including those found in commercial software systems.

To conclude this discussion on the presentation language in LEGAL, it is relevant to compare it with *Spatial SQL* [Ege94]. We have adopted the same principles as *Spatial SQL*: the separation between the query and presentation languages. We consider, however, that *Spatial SQL* is based on the relational model, and therefore is restricted by its limitations; on important problems is the presentation of objects and the values associated to its attributes. By contrast, the proposal outlined in this paper uses the notions of geo-objects, geo-object maps and collections, which allow for object-oriented concepts to be used in a GIS presentation language.

## REFERENCES

- [Ber73] Bertin, J. *Semiologie Graphique*. Paris, Gauthiers-Villars, 1973 (in French).
- [Cat93] Catell, R.G.G. (ed), *The Object Database Standard: ODMG-93*. New York, Morgan-Kaufmann, 1993.
- [CCF96] Câmara, G.; Casanova, M.A.; Freitas, U. "Models, Operations and Languages for Geographical Information Systems". Submitted for publication, 1996. Draft version available in <http://www.inpe.br/spring/legal.html>
- [CFS+94] Câmara, G.; Freitas, U.; Souza, R.C.M., Casanova, M.A.; Hemerly, A.; Medeiros, C.B., "A Model to Cultivate Objects and Manipulate Fields". in *Second ACM Workshop on Advances in Geographic Information Systems*, Proceedings, ACM, Gaithersburg, MD., 1994., pp. 20-27.
- [CSF+96] Câmara, G.; Souza, R.C.M.; Freitas, U.M.; Garrido, J.C.P. "SPRING: Integrating Remote Sensing and GIS with Object-Oriented Data Modelling". *Computers and Graphics*, vol.15, n.6, July 1996.
- [CFO93] Clementini, E.; Di Felice, P.; Van Oosterom, P. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In: *Third International Symposium on Spatial Data Handling. Proceedings*, Singapore, 1993, pp. 277-295.
- [EDH94] Egenhofer, M.; Mark, D.; Herring, J. The 9-Intersection: Formalism and Its Use for Natural-Language Spatial Predicates. *NCGIA Technical Report 94-1*. Santa Barbara, NCGIA, 1994.
- [Ege90] Egenhofer, M. Interaction with GIS via Spatial Queries. *Journal of Visual Languages and Computing*, 1:389-413, 1990.

- [Ege92] Egenhofer, M. Why Not SQL!. *International Journal of Geographic Information Systems*, 6 (2), 1992.
- [Ege94] Egenhofer, M. Spatial SQL: A Query and Presentation Language. *IEEE Transactions on Knowledge and Data Engineering*, 6:86-95, 1994.
- [Goh89] Goh, P-C. A Graphic Query Language for Cartographic and Land Information Systems. *International Journal on Geographical Information Systems*, 1(4):327-334, 1989.
- [Ooi90] Ooi, B. *Efficient Query Processing in Geographical Information Systems*. Springer Verlag Notes on Computer Science, 1990.
- [Mon92] Monmonier, M. *Mapping It Out*. Chicago, Chicago University Press, 1992.
- [SVP+96] School, M.; Voisard, A.; Peloux, J.-P.; Raynal, L.; Rigaux, P., *Geographical DBMS*. Paris, International Thomson, 1996 (in French).