SPATIAL QUERY LANGUAGES: EXTENDED SQL vs. VISUAL LANGUAGES vs. HYPERMAPS

Patrice BOURSIER (1), Michel MAINGUENAUD (2)
(1) Université Paris-Sud - Laboratoire de Recherche en Informatique (URA 410 CNRS)
Bâtiment 490 - 91405 ORSAY - FRANCE
Tél: 33.1.69.41.66.29 Fax: 33.1.69.41.65.86
E-mail: patrice@lri.lri.fr
(2) France Télécom - Institut National des Télécommunications
9, rue Charles-Fourier - 91011 EVRY - FRANCE
Tél: 33.1.60.76.47.82 Fax: 33.1.60.76.47.80
E-mail: maing@int-evry.fr

Abstract

We compare the respective advantages and limitations of three different approaches for querying geographical data bases, namely extended SQL languages, visual languages and hypermaps.
A sample database is defined, along with a typology of queries which is based on a minimal set of queries that a true geographic DBMS should be able to process.

The principles of each of the three approaches are presented. Respective advantages and limitations are pointed out, based on the set of queries previously defined.

1. INTRODUCTION

Designing the user interface is one of the main issues to deal with when building spatial information systems. This is particularly true in the case of Geographic Information Systems (GIS), and it will be more and more true since geographic information is now affecting such general public applications as tourism, and non-specialist users for the management of fleets of vehicles (taxis, fire brigades), or for geo-marketing for example.

Three different approaches may be used for this task:
(i) extend the power of a standard query language, such as SQL,
(ii) use graphical tools to design a visual user interface or query language,
(iii) use hypertext/hypermedia techniques.

The first approach may be considered until now as a research one, since no commercial product fully integrates such complex queries as those involving spatial relationships and deduction. The second approach is also a research one, even if some advanced products have begun to take it into account in the design of user interfaces. The third approach has received few attention until now from the research community.

A toy database is defined in section 2, in order to evaluate the expressive power of the operators available in a GIS. A typology of queries is presented in section 3, which defines in the same time a minimal set of queries that should be handled by a true geographic DBMS. Extended SQL, visual languages and hypermedia approaches are respectively examined in sections 4, 5 and 6. A conclusion is drawn in section 7.

2. SAMPLE DATA BASE

The toy database is defined by eight relations. The attributes are self-explaining except the geometry attribute [Stemple et al., 1986]. The philosophy of abstract data types allows to define an attribute without knowing its physical representation.

In the following, the geometry attribute represents the graphical informations whatever the logical geometrical data model is. The queries we define on this toy database are not concerned at all by the specifications of the geometrical representation [Boursier, 1988; Egenhofer, 1989] (i.e. point, line, polygon) since the geometry is considered as an abstract data type. We retain the relational formalism to define these relations:

Town (#town, name, population, economical_activity, cost_hotel, geometry)
Region (#region, name, main_activity, geometry)
Election_area (#election-area, geometry)
Transport (#transport, name, transport_type, price, geometry)
Network (#line, origin, destination, departure_h, arrival_h, #transport)
Forest (#forest, name, geometry)
Lake (#lake, name, geometry)
Polluted_area (#area, pollution_type, geometry)

The Network relation corresponds to the first level of abstraction of the relation Transport. It allows to see the Transport relation as a logical graph as it is defined in [Mainguenaud, 1991]. In order to simplify the presentation we do not take into account the construction of several relations for the same "layer" (i.e. out of the third normal form).

3. TYPOLOGY OF QUERIES

This section presents seven classes of queries, each of which corresponds to a family. Within a family, sample queries involve new requirements (and therefore a higher expressive power) for the query language.

Non-spatial queries

Query A1 corresponds to a selection in the relational model, query A2 corresponds to a join operation and query A3 corresponds to a "group by + having" clause of SQL[1].

A1 : Which are the towns with more than "100,000" inhabitants ?
A2 : Which towns have the same economical activity ?
A3 : Which are the economical activities that involve more than "100,000 inhabitants" ?

Topology

These queries are concerned with the topological representation of data. Query B1 determines the holes of an "object", query B2 determines the connectivity of an object, and query B3 determines the equality of the geometrical representation of two objects (the geometry must be considered as any other attribute, so this query is equivalent to query A2).

B1 : Which are the regions with some enclaves ?
B2 : Which are the regions with some overseas territories ?
B3 : Which are the regions that correspond to an electoral district ?

---

[1] This query may be more complex in the case of a relational data base system based implementation, due to the constraints of the third normal form.

Spatial relationships

Such queries have been widely studied [Boursier, 1987; Sacks-Davis et al., 1987; Güting, 1988]. Queries C_1 to C_8 respectively involve: distance, geographic localization, inclusion, intersection[2] , ending, join, adjacency and difference.

C_1 : Which towns are situated near a forest ?
C_2 : Which towns are situated in the south of a town named “Paris” ?
C_3 : Which towns are situated in the region named “Ile-de-France” ?
C_4 : Which forests intersect the region named “Ile-de-France” ?
C_5 : Which roads end at the road named “RN13” ?
C_6 : Which is the motorway that continues the motorway named “A6” ?
C_7 : Which towns are bordered by any forest ?
C_8 : Which parts of towns have no intersection with any forest ?

Negation

Query D_1 shows a sample use of this concept in the case of the intersection operator, but it can be generalized to other operators since it can be considered as a unary operator.

D_1 : Which roads do not cross towns with no more than “100,000” inhabitants ?

Disjunction

Query E_1 shows a sample use of disjunction in the case of the intersection operator, but it can also be generalized to other operators. Two restrictions concern the introduction of an aggregate function and the definition of the inclusion operator. Query E_2 implies the possibility to define some selection criteria while evaluating an operator (in order to avoid the cartesian product of the possibilities).

E_1 : Which roads cross towns or forests ?
E_2 : Which roads cross a town, but the road has to be named “RN13” whatever the activity of the town is, or the activity of the town must be “Tourism” whatever the name of the road is‘ [Ooi and Sacks-Davis, 1989]

Aggregation

Aggregation may be defined on the result of an operator (e.g. query F_1 for the intersection operator). It can be generalized to other operators and connectors (e.g. >, < or =). The management of queries F_2 and F_3 presents some specificities that come from the topological representations (without - for query F_2 -, or with - for query F_3 - an overlap of the geometries).

F_1 : Which roads cross one time and only one time the motorway named “A6” ?
F_2 : Which roads have an urban part less than “1 km” long ?
F_3 : Which roads have an intersection with a polluted area, the length of which is less than “1 km” ?

Deduction

Such queries have been widely studied in the database context [Aho and Ullman, 1979]. They mainly require a transitive closure of a relation, here the Network relation. They involve:

- a transitive closure (query G1),
- with selection criteria on the nodes (query G_2),
- with a fixed selection criteria on the node without any order (query G_3),
- with an order (query G_4),
- with negation on the nodes (query G_5),

---

[2] Let Inc_g be the geometrical inclusion of two objects. We assume the specifications of the geometrical intersection operator, Int_g, are defined as: Inc_g (A, B) = A => Int_g (A, B) = A.

- with a selection criteria on the edges (query G6),
- with a fixed selection criteria with an order (query G7),
- with negation on the edges (query G8),
- with a constraint on the result of an operator (query G9, which can be generalized by using other connectors: <, >, ...),
- with a constraint involving the edges (query G10),
- with a constraint on a sub-set of the result from the edge point of view (query G11),
- with a constraint involving at the same time the nodes and the edges (query G12),
- with a constraint on a sub-set of the result (query G13, which can be interpreted as a generalization of the "group by + having" SQL clause),
- with a constraint on a subset of the result from the node point of view (query G14),
- with a constraint involving the nodes (query G15).

These selection criteria and constraints may be considered as a generalization of the "regular expression" defined in [Cruz et al., 1987]. These queries can be considered as examples of one-sided recursion [Naughton, 1987]. The other queries may be defined with many-sided recursion (e.g. query G16 is a well-known query in the field of deductive databases). Query G16 illustrates depth control by the nodes, and query G17 depth control by the edges. The generalization of these queries as the previous regular expression-based queries are restricted to computational queries [Mendelzon and Wood, 89]. Query G18 represents an NP-complete problem [Garey and Johnson, 1979] which can be solved with the application of some heuristics. We do not detail the last queries (G16 to G18) since dealing with these problems in the context of a huge amount of data will obviously lead to very bad performances (and may be inacceptable whatever the hardware is).

G1 : Which routes go from "Paris" to "Nice" ?
G2 : Which routes go from "Paris" to "Nice", with the qualification that visited towns must have more than "100,000" inhabitants ?
G3 : Which routes go from "Paris" to "Nice" with the qualification that visited towns include "Grenoble" and "Valence" ?
G4 : Which routes go from "Paris" to "Nice" with the qualification that visited towns include "Grenoble" and then "Valence" ?
G5 : Which routes go from "Paris" to "Nice" without visiting "Marseille" ?
G6 : Which routes go from "Paris" to "Nice" via the "TGV" train ?
G7 : Which routes go from "Paris" to "Nice" leaving Paris with the "AF" company and arriving at Nice with the "AI" company ?
G8 : Which routes go from "Paris" to "Nice" without using the "AF" company ?
G9 : Which routes go from "Paris" to "Nice" with one and only one visited town ?
G10 : Which routes go from "Paris" to "Nice" in less than "3 hours" ?
G11 : Which routes go from "Paris" to "Nice" with the qualification that in case of connection the departure time is greater than the arrival time ?
G12 : Which routes go from "Paris" to "Nice" with the qualification that the global cost (hotel + transport) is less than "500 FF" ?
G13 : Which routes go from "Paris" to "Nice" with the qualification that the part from "Lyon" to "Marseille" takes the motorway ?
G14 : Which routes go from "Paris" to "Nice" with the qualification that the set of visited towns is the same as the one of the route going from "Lille" to "Nice" ?
G15 : Which routes go from "Paris" to "Nice" with the qualification that the inter-connection time is less than "3 hours" ?
G16 : Which towns are accessible from "Paris" with the same number of visited towns ?
G17 : Which towns are accessible from "Paris" with the same distance[3] ?
G18 : What are the hamiltonian route starting from Paris visiting the towns of "Lyon", "Marseille", "Valence", "Nice" and "Grenoble" ?

---

3 One can remark that this distance is different from the distance of query C1.

The queries presented in this section represent the basic operators. A complex query would involve a combination of these primitives, with a basic assumption: *"the GIS provides operators and not only predicates"*. Predicates actually offer a too weak expressive power if compared to operators.

A sample complex query is defined in order to be expressed with the different query language philosophies:

X1 : Which routes go from "Paris" to "Nice" with the qualification that they border a lake for a while and then cross a town of more than "100,000" inhabitants in its non-forest part".

4. EXTENDED SQL

Principles

Since 1986, SQL has been normalized as a standard query language for relational DBMS. The basic data model is a set of relations, each of which is defined by a set of couples (attribute, domain) where the domain defines the authorized values for an attribute. Obviously, SQL is not suited to query a geographical database. Several propositions have been made in order to extend its expressive power [Bennis et al., 1990; Frank, 1982; Egenhofer M., Frank A., 1988; Güting, 1988; Joseph and Cardenas, 1988; Orenstein and Manola, 1988; Sacks-Davis et al., 1987; Scholl and Voisard, 1989; Svensson and Huang, 1991]. The more complete ones provide operators (in the Select clause), and not only predicates (in the Where clause).

A query is defined as a combination of basic operators. This combination may be performed (i) either by a sequence of SQL statements ("separate approach"), or (ii) by allowing an SQL statement or an operator to be an operand of an operator ("global approach").

Advantages

The main advantage of the SQL language is to be normalized as a relational database query language. It can handle alphanumerical queries (e.g. query A1 to A3). Therefore, it is widely used and its coupling with classical programming languages has been studied for developing applications, even if the implementation is not globally well performed since the syntactic evaluation of an SQL statement is performed at run-time. The basic requirements of the extension by geometrical operators or predicates are generally well accepted (see queries B1 to B3 and C1 to C8).

Drawbacks

Several drawbacks appear with this language: the weak expressive power of the basic relational data model, the optimization of query processing and the management of networks.

The relational data model is based on the third normal form [Codd, 1970]. This constraint is very strong in the case of real data base applications (with alphanumerical data). The extensions, for managing spatial data, are not normalized. In order to avoid the limitations of third normal form, several extended SQL-like languages have been defined on top of an object-oriented data model, but they present an important drawback: they are not normalized. Furthermore, the definition of an attribute by a couple (name, domain) is too weak since no rules are defined to manage the definition of the Select clause (e.g. the attribute "Population of a town" may appear in the Select clause defining the intersection in example X1). No precise rules appear to handle the logical overlap or not of the geometry between two objects of the same "type". This last definition has very important consequences on the geometrical operators (see for example queries F2 and F3).

The optimization of the query evaluation plan is very difficult when a query involves both geometrical and alphanumerical data [Becker and Güting, 1990; Park and Sagev, 1988]. Following the query resolution process, a global optimization of the query must be performed by the DBMS optimizer (the global approach) or outside the DBMS (the separate approach).

The management of network data is not correctly handled in extended SQL query languages (queries G_1 to G_{16}). A network-oriented query may require the definition of a regular expression (query G_7) or constraints on nodes and edges (query G_{12}).

Expression of query X_1

No precise rules are defined to manage the alphanumerical (or the geometrical) results, nor the path operator, i.e. does the path operator appear in the Select clause ? Is it a logical or a geometrical representation ? [Calcinelli and Mainguenaud, 1991]. Example X_1 may be expressed by the following query:

```
Select   *
From     Forest F, Lake L, Network N, Town T
Where    Adjacent (Path ("Paris", "Nice"), L.geometry)
and      Intersection (Path ("Paris", "Nice"), Difference (T.geometry,
           F.geometry))
and      Intersection (F.geometry, T.geometry)
```

One can remark that a combination of spatial operators is required. They may appear in the Where[4] clause or in the Select clause (since part of the town geometry may be relevant for the final result). The definition of such a query in the case of a separate approach may lead to several SQL statements with some uncertainty concerning the global coherence.

```
Insert into New_object
      Select    Difference (F.geometry, T.geometry)
      From      Forest F, Town T
      Where     Intersection (F.geometry, T.geometry)

Insert into Path
      Select    Path ("Paris", "Nice")
      From      Network

Select    *
From      Path P, Lake L, New_object N
Where     Adjacent (P.geometry, L.geometry) and
          Intersection (P.geometry, N.geometry)
```

Following this approach several problems appear, among which the definition of the final result, or the validation of the partial results obtained in New_object and Path (since "objects" appearing in these relations may not satisfy the whole constraints in the query). A global query cannot be performed since the equivalent of the IN statement is not defined for the spatial operators.

5. VISUAL LANGUAGES

Principles

The visual query language concept appeared with the development of "cheap" graphical devices. Propositions of such languages are detailed for example in [Angelaccio et al., 1990; Chang, 1987; Kim et al., 1988; Kuhn, 1990; Shu, 1988]. A query language is said to be visual whenever the semantics of the query is expressed by a drawing. Following the philosophy of classical programming languages, two main directions may be defined: (i) imperative query languages, and (ii) declarative query languages.

---

[4] To simplify the presentation, we do not take into account geometrical properties in the Where clause (e.g. minimum bounding rectangle), which can be defined as attributes or as an internal information of an abstract data type representation used by the geometrical operator.

Imperative query languages define step by step the elementary operations. [Kirby and Pazner, 1990] gives an example of such a language.

Declarative query languages define the properties to be verified by the final result. [Mainguenaud and Portier, 1990] gives an example of such a language. To evaluate a query, the first step is to convert this drawing into a formal expression. The second step is to convert this expression into DBMS understandable orders.

Advantages

Two main advantages are provided by these languages: (i) they are more natural, and (ii) they allow the combination of operations.

The first main advantage of visual languages is to be more natural to the end-user. The basic components are graphical and they reflect the two dimensions of the objects they model (e.g. query C_1 to C_8). The data model may be "object-oriented" at the top level (even if it is not the case at the DBMS level). The notion of combination is inherent to the querying philosophy. This functionality represents the second main advantage since the query resolution model is therefore application-independent (see example X_1).

Drawbacks

Three main drawbacks appear with visual languages: lack of normalization, complexity of a query and expression of negation.

Two levels may be defined while using such languages: (i) the graphical level, and (ii) the logical formalism for modelling a query. The first level is not normalized at all since this field is very new. The logical formalism may be an SQL-like language if the combination of operators is allowed, otherwise a new formalism has to be defined. This logical formalism can be the level of interface between applications developed with a classical programming language (e.g. Lisp, C or Pascal) and the GIS. Since the level of interface with the DBMS is not normalized, these languages have the same limitations than Extended SQL languages.

In the case of a query with a lot of spatial constraints defined between inter-related objects, the graphical representation may be very confusing. Furthermore, the 2D representation may involve some ambiguities in the graphical representation [Calcinelli and Mainguenaud, 1991].

The philosophy of querying by example (with the semantics of the operators and not by the semantics of the data model -i.e. relations-) leads to provide an example of the spatial relationships. Therefore the negation concept is difficult to express in the context of 2D since the relationship does not exist by definition -an ambiguity may appear between two independent "objects"- (see for example query C_1).

Expression of query X_1

To illustrate such languages, we adopt the formalism of CIGALES [Mainguenaud and Portier, 1990]. Example X_1 is illustrated in Figure 1.

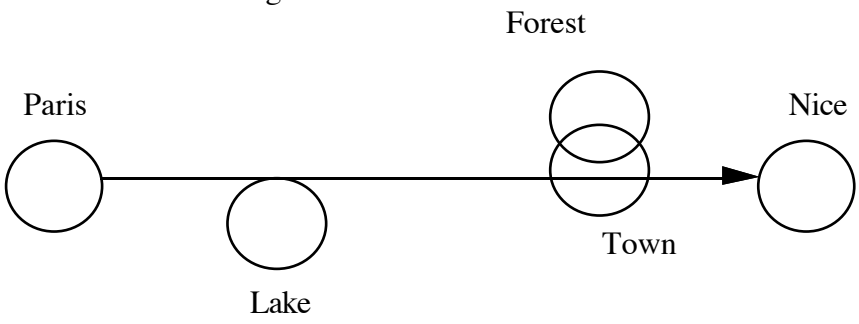

Figure 1: Visual language expression of query X_1.

6. HYPERMEDIA

Principles

Hypermedia is a relatively young technology, and real and significant applications are still very few. The “hypertext” concept dates from 1945, with Bush’s “Memex”[Bush, 45]. More recently, Conklin has proposed a review and synthesis of the subject [Conklin, 87].

The basic feature of hypertext and hypermedia systems is to let users “navigate” through a set of information with the help of **links**. These links are represented on the screen by **buttons** appearing as icons/pictograms or typographical enrichments in text.

Until now, the hypermedia concept has not been much considered in the field of geographic information management and processing [Gaines, 1988]. We have used this concept to develop a CD-ROM based electronic atlas of Europe (limited to countries that belong to the European Economic Community) [Boursier, 1990].

Advantages

The main advantage of what we may call “hypermaps” is due to the nature of hypermedia technology. No query language has to be known, the only thing the user needs to know is the meaning of buttons that appear on the screen. No query language can be so simple, since the user just shows what he wants.

An other advantage of hypermaps comes from the multi-media orientation of hypermedia, that allows to link parts of text, vector-based geographic entities, raster-based maps or images, or even sounds or video [Boursier 1991].

Drawbacks

The main limitation of hypermedia systems is due to the fact that queries must have been previously defined and “prepared”, i.e. links must have been established between entities. In other words, paths must have been pre-compiled, and the user cannot ask for anything else.

The other main drawback comes from the lack of an underlying data model. The only predefined concept is the use of links between entities.

Expression of query X_1

Clearly, query X_1 cannot be expressed with hypermaps, unless a button “X_1” be displayed on the screen, thus indicating that a link exists either directly to the answer if the corresponding map (or picture) is available, or indirectly if a portion of code is associated to the button.

7. CONCLUSION

We have compared the respective advantages and limitations of three different approaches for querying geographical data bases, namely extended SQL languages, visual languages and hypermaps.

A sample database has been defined, along with a typology of queries which is based on a minimal set of queries that a true geographic DBMS should be able to process.

An extended SQL query language seems to be the best way to define a database query at the DBMS level. Nevertheless, it needs to be normalized and to be augmented with several functionalities in order to handle network processing and with the specifications on the data model (i.e. properties still true on a sub-set of the "object", the overlap or not of the geometry). The development of an application may be performed at the extended SQL level if this language accepts the composition of spatial operations in the Select clause (global approach). In the case of a separate approach, a higher

level is required so as to ensure the global coherence since the resolution mechanisms are independent from an application (the query resolution model is always the same).

Visual programming languages are very promising since they are more natural than extended SQL statements. Nevertheless, since the level of abstraction is higher, the interaction with the DBMS is more complex. This difference of level leads to define a formal expression to model a query which may have a higher expressive power than SQL statements provide.

Hypermedia-based geographic information handling is interesting because it is closer to a naive user way of thinking. For that reason, it has been used mainly for general public applications on interactive terminals. For that reason also, the expressive power of hypermedia-based querying is limited. The use of "hypermaps" cannot really be considered as the use of a query language since queries are restricted to what has been previously defined in terms of links between entities. It is more navigation inside some kind of semantic network than real and open querying.

We believe that combining these different approaches could lead to spatial query languages that would be more powerful, less dependent on a specific data model, and easier to use at the same time. We believe in particular that it would be worth mixing visual language and hypermedia approaches at the user/external level, but it would also be very interesting to have a standard extended SQL language available at the DBMS level.

REFERENCES

Aho A., Ullman J.D., 1979. "Universality of Data Retrieval Languages", Proceedings of the ACM Principles Of Programming Languages Conf., San-Antonio, USA, January 1979.

Angelaccio M., Catarci T., Santucci G., 1990. "QBD*: A Graphical Query Language with Recursion", IEEE Trans. On Software Engineering (TOSE), Vol. 16, No. 10, October 1990.

Becker L., Guting R.H., 1990. "Ruled-Based Optimization and Query Processing in an Extensible Geometric Database System", Technical Report Universität Dortmund, 1990.

Bennis K., David B., Quilio I., Viemont Y., 1990. "Geotropics: Database Support Alternatives for Cartographic Applications", 4th Int. Symposium on Spatial Data Handling, Zürich, Switzerland, July 1990.

Boursier P., 1987. "Application Planification Urbaine (Urban Planning Application)", Altaïr Technical Report, No. 9-87 (in french), GIP Altaïr, Rocquencourt, France, 1987.

Boursier P., 1988. "Analysis of Urban Geographic Queries", Computer Graphics International Conf., Geneva, Switzerland, May 1988.

Boursier P., et al., 1990. "EUROMAP: a CD-ROM based Digital Cartographic Atlas of Europe", 1st European Conference on Geographical Information Systems (EGIS' 90), Amsterdam, The Netherlands, April 1990.

Boursier P., 1991. "Hypermedia-based geographic information handling", 2nd European Conference on Geographical Information Systems (EGIS' 91), Brussels, Belgium, April 1991.

Bush, 1945. Bush V., "As we may think", Atlantic Monthly, July 1945.

Calcinelli D., Mainguenaud M., 1991. "The Management of Ambiguities in a Graphical Query Language for Geographical Information Systems", 2nd Symposium on Large Spatial Databases, Zürich, Switzerland, August 1991 (Lecture Notes in Computer Sciences, No. 525).

Chang S.K., 1987. "Visual languages: a tutorial and survey", IEEE Software, January 1987.

Codd E.F., 1970. "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, Vol. 13, No. 6, June 1970.

Conklin, 1987. Conklin J., "Hypertext: an introduction an survey", IEEE Computer, September 1987.

Cruz I.F., Mendelzon A.O., Wood P.T., 1987. "A Graphical Query Language Supporting Recursion", Proceedings SIGMOD Conf., San-Fransisco, USA, May 1987.

Egenhofer M., Frank A., 1988. "Towards a Spatial Query Language: User Interface Considerations", 14th Int. Conf. on Very Large Databases (VLDB' 88), Los Angeles, USA, August 1988.

Egenhofer M., 1989. "A Formal Definition of Binary Topological Relationships", 3rd Int. Conf. on Foundations of Data Organization and Algorithms, Paris, France, June 1989.

Frank A., 1982. "MAPQUERY: Database Query Language for Retrieval of Geometric Data and their Graphical Representation", Computer Graphics, Vol. 16, No. 3, July 1982.

Gaines, 1988. Gaines B., Vickers J., "Hypermedia design", RIAO '88 Conf. on User-Oriented Content-Based Text and Image Handling, Cambridge, MA (Etats-Unis), 1988.

Garey M.R., Johnson D.S., 1979. "Computer and Intractability - A guide to the theory of NP-completeness", W.H. Freeman and Co., New-York, 1979.

Güting R.H., 1988. "Geo-Relational Algebra: A Model and Query Language for Geometric Database Systems", Proceedings of Int. Conf. on Extending Data Base Technology, Venice, Italy, March 1988.

Joseph T., Cardenas A.F., 1988. "PicQuery: A High Level Query Language for Pictorial DBMS", IEEE TOSE, Vol. 14, No. 5, May 1988.

Kim H.Y., et al., 1988. "PICASSO : A Graphical Query Language", Software -Practice and Experience, Vol. 18, No. 3, pp. 169-203, March 1988.

Kirby K.C., Pazner M., 1990. "Graphic Map Algebra", 4th Int. Symposium on Spatial Data Handling, Zürich, Switzerland, July 1990.

Kuhn W., 1990. "Editing Spatial Relations", 4th Int. Symposium on Spatial Data Handling, Zürich, July 1990.

Mainguenaud M., Portier M.A., 1990. "CIGALES: A Graphical Query Language for Geographical Information Systems", 4th Int. Symposium on Spatial Data Handling (SDH), Zürich, July 1990.

Mainguenaud M., Raffy J.L., Simatic X.T., 1991. "Graph Manipulations for Network Oriented Data Management - Application to a Telecommunication Network", 14th Urban Data Management Symposium (UDMS), Odense, Denmark, May 1991.

Mendelzon A.O., Wood P.T., 1989. "Finding Regular Paths in Graph Databases", Proceedings of VLDB Conf., Amsterdam, The Netherlands, July 1989.

Naughton J.F., 1987. "One-sided Recursion", Proceedings of the 6th PODS Conf., San-Diego, USA, March 1987.

Ooi B.C., Sacks-Davis R., 1989. "Query Optimization in an Extended DBMS", 3rd Int. Conf. on Foundations of Data Organization and Algorithms (FODO), Paris, France, June 1989.

Orenstein J.A., Manola F.A., 1988. "Probe Spatial Data Modelling and Query Processing in an Image Database Application", IEEE TOSE, Vol. 14, No. 5, May 1988.

Park J., Sagev A., 1988. "Using Common Sub-Expression to Optimize Multiple Queries", Proceedings of the Int. Conf. on Data Engeneering, Los Angeles, USA, February 1988.

Sacks-Davis R., Mc Donnell K.J., Ooi B.C., 1987. "GEOQL: A Query Language for Geographical Information Systems", Australian and New Zealand Ass. for the Advancement of Science Congress, Townsville, Australia, August 1987.

Scholl M., Voisard A., 1989. "Thematic Map Modelling", 1st Int. Symposium on Large Spatial Databases, Santa Barbara, USA, July 1989 (LNCS n°274).

Shu N.C., 1988. "Visual Programming", Van Nostrand Reinhold Cy, New York, 1988.

Stemple D., Sheard T., Bunker R., 1986. “Abstract Data Types in Databases: Specification, Manipulation and Access”, Proceedings of Int. Conf. on Data Engeneering, Los Angeles, USA, February 1986.

Svensson P., Huang Z., 1991. “Geo-SAL: A query Language for Spatial Data Analysis”, 2nd Symposium on Large Spatial Databases, Zürich, Switzerland, August 1991 (LNCS n°525).