1988
14th VLDB

**Los Angeles
USA**
**Aug. 29/
Sept. 1**

PROCEEDINGS OF THE FOURTEENTH INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES

Editors: Francois Bancilhon & David J. DeWitt

Towards a Spatial Query Language: User Interface Considerations*

Max J. Egenhofer
Andrew U. Frank
Surveying Engineering Program
University of Maine
Orono, ME 04469, USA
MAX@MECAN1.bitnet
FRANK@MECAN1.bitnet

Abstract

Spatial information systems are large collections of spatial and non-spatial data on top of tailored database management systems with interactive analysis being the rule rather than an exception. Current database query languages, which are well-suited to treat alphanumeric data, do not reflect the properties of spatial data. In order to visualize the power needed for a spatial query language, a user interface is designed which gives specific considerations to the coexistence of representations as graphical renderings, such as maps, and lexical listings, such as tables. The different properties of spatial and non-spatial data give rise to lexical formulations of queries in combination with references to graphical 'objects' or areas on maps. In particular, immediate reference to objects on drawings with direct manipulation devices is a crucial technique to be embedded in the interaction. Variation of graphical presentations by colors, patterns, etc. requires appropriate tools in the interface to manipulate the presentation of spatial objects. These issues are visualized in a sample session by a sequence of interface snapshots.

1 Introduction

A Geographic Information System (GIS) or a Land Information System (LIS) is a heterogeneous collection of

spatial and non-spatial data managed in a computerized system. The collection comprises such vastly different domains as geometric and topographic data resulting from surveys, geographic descriptions, legal aspects, projects of urban and regional planning, etc. Due to its multi-purpose character, a Geographic Information System is designated to be useful for a large variety of users, sharing fundamental data, such as topography, and adding their individual project data.

Traditionally, a spatial information system is designed on top of a database which must be tailored to the specifics of spatial data [Frank 1981] [Frank 1984a] [Härder 1985]. While storage- and access-management of spatial data in such a system have been explored for some time [Frank 1983] [Guttman 1984] [Nievergelt 1984] [Orenstein 1986] [Tamminen 1982], little attention has been paid to the design of user interfaces for spatial information systems. The demands for such interfaces differ significantly from those for database applications with exclusively alphanumeric data. This paper investigates how the properties of spatial data, in particular their graphical representation, contribute to a hybrid interaction between user and system. The design of query languages to deal with spatial data is expected to benefit from these considerations. In particular, the integration of tools, such as direct manipulation devices, pull-down menus, and control panels, into the dialogue are essential to have a powerful, 'fluent' language.

The paper starts with an investigation of dialogue in mixed spatial/non-spatial systems. In the subsequent section, the environment for presentations of spatial data is discussed. Varying graphical representation and selection of appropriate context are presented as powerful techniques to manipulate spatial data. The last section demonstrates how a user interacts with a combined spatial/lexical system, simulating a sample session of a Geographic Information System. A sequence of interface snapshots illustrates the users' view and their actions.

2 Integrating Lexical And Graphical Queries

The interface of a spatial information system is the integrating part of all particular applications combining graphical and lexical representation, as well as the query language. Users articulate their instructions through a dedicated language to communicate with the system. This language must include tools for all the essential operations.

The Seeheim model of interfaces [Green 1985] differentiates between three parts of a User Interface Management System: (1) the presentation, (2) the dialogue, and (3) the application interface. Conventional database query languages are restricted to the treatment the properties of non-spatial data. GIS-typed user interfaces differ from conventional information systems by various graphical representations of spatial objects, and the specific interaction between spatial and non-spatial data. Both issues require a thorough understanding in order to be appropriately embedded in the user interface. This section presents three major types of interaction that must be integrated into a user interface for a Geographic Information System: (1) lexical conversation, (2) interaction with graphically displayed results, and (3) mixed lexical-graphical conversation.

2.1 Alphanumeric Query Languages

Structured query languages for question/answer-dialogue dominate in conventional, purely alphanumeric information systems. Well-known examples are SQL [Chamberlin 1976], QUEL [Stonebraker 1976], and Query-By-Example [Zloof 1977]. Similar languages with extended capabilities for treating spatial objects and properties, such as MAPQUERY [Frank 1982], enhanced with powerful 'expert system' tools, such as LOBSTER [Frank 1984b] or KBGIS [Smith 1986], are expected to be the framework for the interaction with future spatial information systems.

A typical example for a purely lexical query in a GIS environment is "How far is it from Boston to Orono on the Interstate I-95". Such queries are characterized by alphanumeric input and results which are representable as tables or similarly structured compositions. Conceptually, these queries can be formulated, processed, and answered with the type of conventional query languages mentioned above; however, extensions are needed to incorporate into the language the specific properties of spatial data, such as relations among spatial objects [Egenhofer 1987].

2.2 Queries With Graphical Results

The most obvious difference between conventional and spatial systems is the ability to present spatial data graphically. Interactive-graphic presentation is powerful for mapping systems, because the content of maps can be quickly modified. While traditional mapping is a static product—once the map is printed it cannot be updated unless overwriting it with some pencil—mapping with spatial information systems promotes immediate manipulations on drawings. Objects can be added to, removed from, or modified on an existing map without the need to start with a new drawing from scratch again. This implies that the user must have tools to manipulate a map. Unlike conventional systems which treat each result as an entirely new representation and do not refer to earlier results, several interactive-graphic drawings are often overlayed over each other or one 'thematic layer' is removed from another. These processes are the particular power of spatial information systems, combining spatial data of arbitrary sources to a composition that appears most suitable for a user's application. Furthermore, individual objects on a map may be highlighted to facilitate their identification in complex situations. Imagine a town map with streets and buildings on it, and the user wants to find the City Hall. Instead of checking each building, it is more convenient and faster to choose a specific presentation for the City Hall that makes it easy to distinguish from other buildings. Therefore, an interactive-graphic information system needs four graphical presentation types:

- *New* refreshes the viewport before drawing the next picture.

- *Overlay* adds the result of the current query to the existing picture.

- *Remove* erases the result of the current query from the existing picture.

- *Highlight* presents the result in a fashion that draws the user's attention to it.

Of course, conventional systems with lexical presentation could adopt these presentation types, though tabular presentation appears to be less complex and more statically oriented than graphical presentation. It is worthwhile to investigate how users may benefit from tabular presentation with dynamic updates.

2.3 Lexical-Graphical Interaction

The coexistence of graphical and lexical data leads to a hybrid interaction in which lexically formulated queries are enhanced by pointing to objects on a drawing. Graphical renderings, such as maps, animate to interpret points,

125

lines, and areas as objects, and to use direct-manipulation devices, such as mice, light-pens, etc. to refer to them as 'objects'. Direct-manipulation [Shneiderman 1983] has been promoted as an efficient way to refer to representations on a screen in which the interaction between the user and the system is geared towards 'seeing and pointing' in lieu of 'remembering and typing' [Smith 1983].

Two distinct types of referencing graphical representations must be supported in a query environment for Geographical Information Systems, (1) selection of sub-areas for more detailed representations, and (2) identification of individual objects via *pick*.

2.3.1 Area Selection

On maps, the same object can be presented at different levels of detail and in different scales [Beard 1987]. This multiple presentation of spatial objects, called cartographic generalization, is frequently used in Geographic Information Systems to suppress or expand details. Typically, a situation is first looked at from a greater distance (small scale representation) before moving stepwise closer toward the region or object of interest (large scale representation). This process of iteratively approaching a graphical scene—using the current rendering as reference—must be enbedded into the dialogue and be immediately supported by direct-manipulation methods.

The selection of subareas needs four interaction steps, (1) the prompt for the user's selection, (2) verification of the user's input, (3) possible corrections, and (4) confirmation of the selection by the user.

An efficient method for selecting a part on a map is to choose a rectangle with its edges parallel to the screen axes which delineate the new view. With a direct-manipulation device, the rectangle can be selected by pointing at two of its opposite corners. The rectangular approach is most often sufficient to determine an area for zooming; however, other irregular shapes may be needed to support queries that refer to parts of objects.

Zooming into a drawing is often linked to the users' desire for even-scaled maps, for instance, 1:10,000 is preferred over 1:9,972.32. The selections of scale and subarea can be linked together in the interface. Because the range of the viewport is determined by the scale, a subarea can be easily selected by dragging a scaled window over the desired location. This technique gives the users a good impression of the range they are going to see.

2.3.2 Selection of Objects by Pointing

Users prefer to reference objects by pointing at them rather than typing some identification. A building on a map, for example, is more conveniently referenced by pointing at it instead of using the address of the building

as identification. A typical GIS-typed query with direct graphic input is "Who owns this building?", and the user refers to a building on a map. Selection by pointing is a major object-oriented feature of graphical presentation since users can select what they see.

The interaction must be supported by some dialogue leading the user through the possibly complex process of selecting objects by pointing on them. The dialogue is initiated by the lexical input containing keywords for selecting by pointing. The following interaction processes must be supported in the interface:

- The user is prompted indicating when to select the object(s). The prompting must provide the user feedback about which object types to select, as well as activate the pointing device. This help is especially important for queries with multiple *picks* on different object classes. Ideally, the query as typed is used in this communication, highlighting the object types of the current selection.

- The objects selected are immediately verified so that the users have control over their selections and what the system actually identified. The particular identification of the target is a separate process, outside of the interaction, and will not be discussed here.

- Before confirming the selection, the users can, at any time, alter their choice by cancelling previously selected objects. Such corrections must include the corresponding verification which resets the presentation of the cancelled objects to their original state.

- Confirmation is the final user action. 'Double clicking', pressing a dedicated button on a multi-button mouse, or using the return key on the keyboard are commonly used methods for confirmation.

3 Environment

Graphical representation is more complex and richer in variants than any lexical representation. Three significantly different concepts can be identified: (1) varying representation of objects by using colors, patterns, intensity, and symbols [Egenhofer 1988], (2) the legend as the key for the interpretation of the map, and (3) selection of context as background [Frank 1982].

3.1 Varying Graphical Presentations

In GIS applications, often the same data is presented in various fashions according to different views. The view of a road, for example, is different for a surveyor, a transportation company, or a cartographer; the one sees it as

126

a parcel, the other as a classified transportation line, another as a symbolic line, and for each purpose the road must be presented differently. Such varying presentation is different from generalization. While generalization establishes different abstraction levels for an object class [Brodie 1984], varying presentation relies upon the same geometry of an object, and alters only the appearance of the object with tools like colors and patterns.

The choice of the presentation must be controlable from the interface to allow the users to adopt the most appropriate presentation type for their application. This flexibility is an important contribution to the multi-purpose applications of a single data collection, and the user interface must provide tools to accomodate the presentation to the individual, often subjective, needs. Four object-oriented methods, based on the purely graphical properties of Bertin [Bertin 1983], are distinguished to vary the graphical representation of an object: (1) colors, (2) patterns, (3) intensity, and (4) symbols.

3.2 Context

The interpretation of a graphical representation is extremly sensitive to the context and environment in which it is shown. Unlike lexical representation, it is often not sufficient to present exclusively what was asked for. Imagine a query "Show the town of Orono", where the result is only a labeled point in the middle of the screen [Frank 1982].
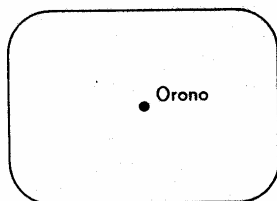
A reasonable answer would have required a context in which the position of the point could have been spatially interpreted. For example, by showing the borders of the state of Maine, sufficient information is given to locate Orono.
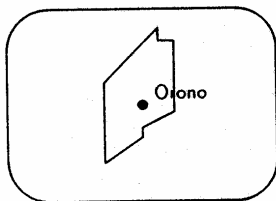
Sophisticated graphical representation considers the selection of appropriate context which depends upon the purpose of the drawing, the scale, and the data density. The users must be provided tools to select the most efficient context for their maps immediately from the user interface.

3.3 Legend

The legend is the key for the interpretation of a drawing with varying graphical presentation. For each variant used in the drawing it lists the corresponding meaning. Without a legend, symbols are sometimes difficult to interpret. For example, a map with four different symbols for cities, each presenting a different range of size, requires a legend like the following stating the otherwise not transparent meanings:

○	city less than 10,000 population
◉	city between 10,000 and 50,000 population
●	city between 50,001 and 100,000 population
⬤	city greater than 100,000 population

The legend is specific for a map, and each graphical representation can have a separate legend. The interaction with the legend asks for a flexible design: (1) The legend must be accessible immediately from the user interface whenever information is required. Due to the limited screen area it is not possible to have the whole legend permanently visible. (2) The interface must support the desire to visually compare the legends of two maps.

3.4 Layout of the Interface

The layout of the interface must support the users' analyses. Sites, trails, and modes, informing the users about their current, previous, and forthcoming states, are important hints [Nievergelt 1979], often considered in conventional systems. Additional guidance is desirable if spatial data is displayed. In order to analyze an image it is important to know what was actually represented and what was suppressed. A compilation of all queries which were asked to compose the current drawing is an important information for the user. A flashback through the former queries is often only helpful if the query and the result are concurrently visible.

Qualitative analysis is often based upon visual comparison of several situations, an issue which is difficult unless the situations to be compared are visible at a time. The interface must be layed out such that either two maps or a map and a lexical result are concurrently visible.

127

The general desire to arrange all results so that they are always comparable is drastically restricted by the limited screen size. Even overlapping windows which appeared to be well-suited for this type of representation turned out to complicate analyses significantly [Bly 1986]. Rather, a tiled interface with a limited number of concurrent windows seems to help the users concentrate on their jobs, namely extracting information from the system, and discourage them from 'playing' with the window arrangement [Newman 1983].

## 4 A Guided Tour Through An Interface

The interaction between users and system is often hard to imagine unless the users experienced the dialogue by themselves in an actual session. In this section, the interaction with a Geographic Information System is simulated by a sequence of snapshots of the user interface documenting the flow of interaction between the user and the system. This technique will give a vital impression of the concepts previously introduced, and is extremely helpful during the design phase of interfaces.

The interface is subdivided into five major parts, (1) the alphanumeric input area at the bottom, (2) the lexical output area to the right, (3) the graphical output area in the center, (4) the control panel to the left, and (5) the pull-down menus for the setup. The lexical and graphical output areas can be subdivided into smaller, independent areas which can be individually addressed. On top of each viewport, a horizontal bar (6) contains a control panel to evoke timely information about the context and legend of the particular drawing.
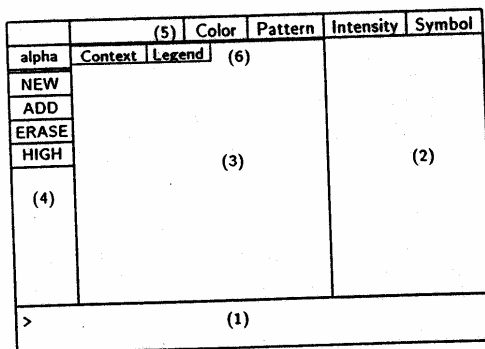
The commands are entered from the keybord in an extended SQL language [Egenhofer 1987] which was customized for the properties of spatial objects and interactive communication. By default, making a NEW drawing is the initial output mode. Typically, a GIS session starts with a query for an overview map of the area of interest,

e.g., a map of the state of Maine with all highways. In order to have the highways drawn in a specific symbol discriminating them from the state border line, the user sets up the appropriate symbol for the highway.
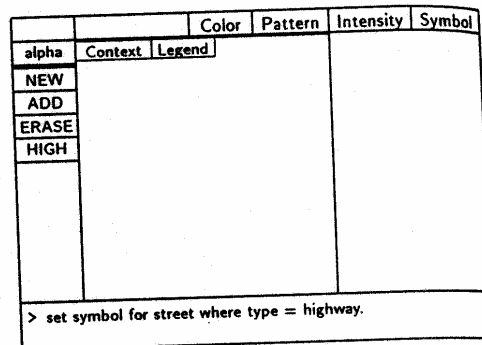
The particular symbol is selected from the symbol pull-down menu.
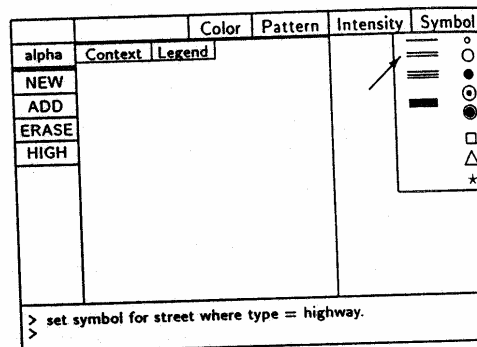
The setup can be verified by checking the legend in wh the double-lined symbol shows up for the highway.

Before starting with the query, the graphical context of the map is checked which will be added to all graphical representations.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW	state where name = 'Maine'				
ADD					
ERASE					
HIGH					
>					

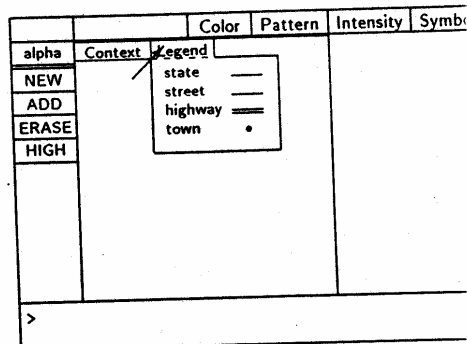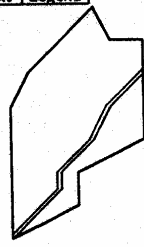It shows that the border line of the state will be added to each drawing without being explicitly specified in the query. Now the user selects **NEW** from the mode control panel to draw a new map and enters the query for all highways.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					
> select geometry from street where street.type = 'Highway'.					

As result, the state border and the highways show up on the graphic display.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					
> select geometry from street where street.type = 'Highway'.					
>					

With the next query, the user wants to have the cities overlayed over the current map. So, **ADD** is selected from the control panel, and then the query requesting all towns is entered from the keyboard.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					
> select geometry from town.					

As result, all towns are displayed on the current map.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					
> select geometry from town.					
>					

This indifferent representation is confusing. In fact, the user wanted to see only the largest cities, not all of them. To remove all cities with less than 10,000 population, **ERASE** is set on the control panel, and the query for all cities with less than 10,000 population is entered.

129

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERAS					
HIGH					

> select geometry from town where population < 10,000.

As result, only the cities with 10,000 inhabitants or more remain on the map.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERAS					
HIGH					

> select geometry from town where population < 10,000.
>

On this map, the user would like to have shown the location of Bangor. So, HIGH is selected to highlight the query result, and then the query for the town Bangor is entered.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					

> select geometry from town where name = 'Bangor'.

The location of Bangor is highlighted on the map.

Now the user wants to get the name of the city next to Bangor. alpha is selected from the control panel for the alphanumeric mode, and then the query for the name of the town identified by pick is entered.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					

> select geometry from town where name = 'Bangor'.
>

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					

> select name from town where geometry = pick.

The cursor shows up prompting the user for input with the mouse. The highlighted keyword town indicates the object type to select.

		Color	Pattern	Intensity	Symbol
alpha	Context Legend				
NEW					
ADD					
ERASE					
HIGH					

> select name from town where geometry = pick.
>

The user moves the cursor to the desired city and presses the button. After a few moments, the city gets highlighted

130

verifying that this object was identified. The identfication is confirmed by clicking twice at the mouse button, and the name of the town appears in the alphanumeric window.

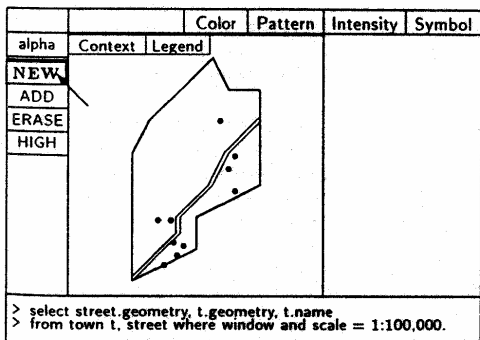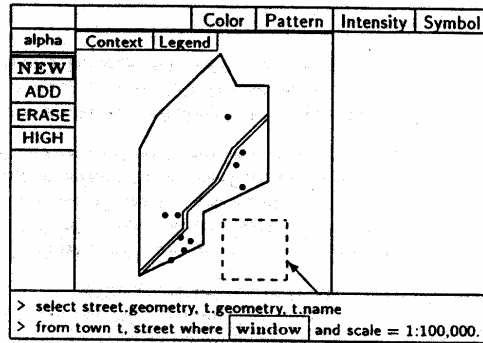|		Color	Pattern	Intensity	Symbol
alpha	Context	Legend			
NEW				town.name	
ADD					
ERASE				Orono	
HIGH					

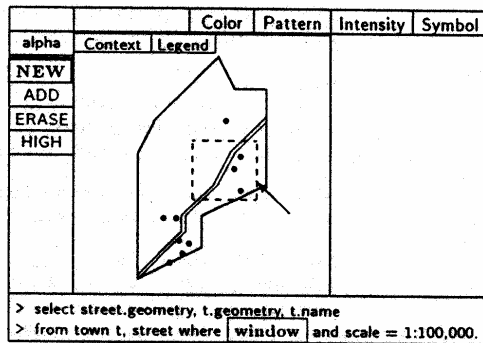> select name from town where geometry = pick.
>
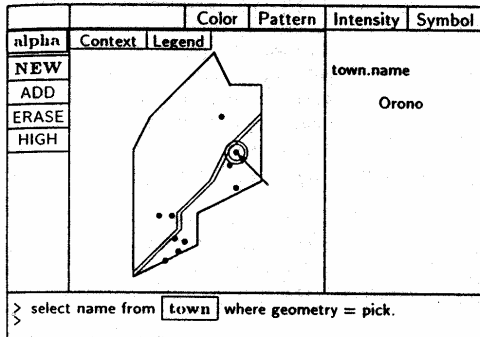
To get more detailed information about this area, the user asks for a larger-scaled map. First, NEW is selected to produce a new map, and then the query for all towns and streets is entered, specifying a window in 1:100,000. Note that the town names as non-graphical properties will be added as labels into the drawing.
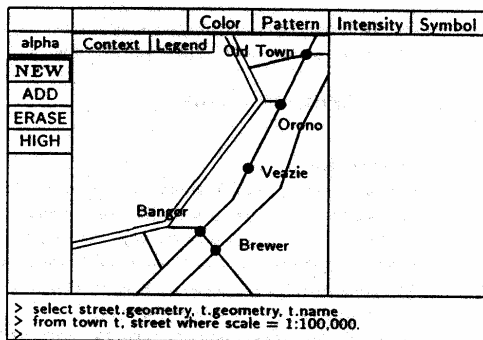
		Color	Pattern	Intensity	Symbol
alpha	Context	Legend			
NEW					
ADD					
ERASE					
HIGH					

> select street.geometry, t.geometry, t.name
> from town t, street where window and scale = 1:100,000.

A window, marking a subarea in 1:100,000, shows up on the screen. The highlighted window keyword in the query indicates the action to take.

		Color	Pattern	Intensity	Symbol
alpha	Context	Legend			
NEW					
ADD					
ERASE					
HIGH					

> select street.geometry, t.geometry, t.name
> from town t, street where window and scale = 1:100,000.

The user moves the window over the desired area and clicks on the mouse button.

		Color	Pattern	Intensity	Symbol
alpha	Context	Legend			
NEW					
ADD					
ERASE					
HIGH					

> select street.geometry, t.geometry, t.name
> from town t, street where window and scale = 1:100,000.

The result is a more detailed map of the subarea.

		Color	Pattern	Intensity	Symbol
alpha	Context	Legend	Old Town		
NEW					
ADD				Orono	
ERASE					
HIGH			Bangor	Veazie	
				Brewer	

> select street.geometry, t.geometry, t.name
> from town t, street where scale = 1:100,000.
>

5 Conclusion

The design of user interfaces plays a vital role for spatial information systems. The interaction with spatial data

131

differs significantly from the interaction with purely lexi-
cal data. Query languages for spatial information systems
must incorporate techniques to treat the specific proper-
ties of spatial objects. This paper identified the graphical,
object-oriented representation of spatial data as a crucial
issue demanding for various display types, such as overlay,
erase, and highlight. Interaction with graphically repre-
sented spatial objects requires methods, such as selection
by pointing to objects and selection of subareas. The en-
vironment specific for representation of spatial data needs
selection of context and tools to manipulate the graphical
representation. The concepts were visualized in a series
of interface snapshots. A prototype of the query language
has been partially implemented and will be integrated into
the proposed interface.

6 Acknowledgement

Thanks to Cathy Page, Mark Palmer, and Jeff Jackson
for their comments on the layout of the interface.

References

[Beard 1987] K. Beard. How To Survive A Single Detailed
Database. In: N.R. Chrisman, editor, Eighth
International Symposium on Computer-Assisted
Cartography, Baltimore (MD), 1987.

[Bertin 1983] J. Bertin. Semiology of Graphics. The Uni-
versity of Wisconsin Press, Madison (WI), 1983.

[Bly 1986] S.A. Bly and J.K. Rosenberg. A Comparison
of Tiled and Overlapping Windows. In: Human
Factors in Computing Systems, CHI '86, April
1986.

[Brodie 1984] M.L. Brodie. On the Development of Data
Models. In: M.L. Brodie et al., editors, On
Conceptual Modelling, Springer Verlag, New
York (NY), 1984.

[Chamberlin 1976] D.D. Chamberlin et al. SEQUEL 2:
A Unified Approach to Data Definition, Manip-
ulation, and Control. IBM Journal of Research
and Development 20(6), 1976.

[Egenhofer 1987] M. Egenhofer. An Extended SQL Syn-
tax To Treat Spatial Objects. In: Y.C. Lee, ed-
itor, Proceedings of the Second International
Seminar on Trends and Concerns of Spatial Sci-
ences, Fredericton (New Brunswick), 1987.

[Egenhofer 1988] M. Egenhofer. Graphical Representa-
tion of Spatial Objects: An Object-Oriented
View. In: Third International Conference on

Engineering Graphics and Descriptive Geometry,
Vienna (Austria), July 1988.

[Frank 1981] A. Frank. Applications of DBMS to Land
Information Systems. In: C. Zaniolo and C. De-
lobel, editors, Seventh International Conference
on Very Large Data Bases, Cannes (France),
1981.

[Frank 1982] A. Frank. MAPQUERY—Database Query
Language for Retrieval of Geometric Data and
its Graphical Representation. In: D. Bergeron,
editor, SIGGRAPH '82, ACM Computer Graph-
ics, Boston (MA), July 1982.

[Frank 1983] A. Frank. Problems of Realizing LIS: Stor-
age Methods for Space Related Data: The Field
Tree. Technical Report 71, Institut for Geodesy
and Photogrammetry, Swiss Federal Institute of
Technology (ETH), Zurich (Switzerland), 1983.

[Frank 1984a] A. Frank. Requirements for Database
Systems Suitable to Manage Large Spatial
Databases. In: International Symposium on
Spatial Data Handling, Zurich (Switzerland),
August 1984.

[Frank 1984b] A. Frank. Extending a Database with Pro-
log. In: L. Kerschberg, editor, Proceedings
of the First International Workshop on Expert
Database Systems, Kiawah Island (SC), Octo-
ber 1984.

[Green 1985] M. Green. Report on Dialogue Specifica-
tion Tools. In: G.E. Pfaff, editor, Computer
Graphics Forum, Springer Verlag, 1985.

[Guttman 1984] A. Guttman. R-Trees: A Dynamic In-
dex Structure for Spatial Searching. In: Pro-
ceedings of the Annual Meeting ACM SIGMOD
Boston (MA), 1984.

[Härder 1985] T. Härder and A. Reuter. Architecture
of Database Systems for Non-Standard Applica-
tions (in German),. In: A. Blaser and P. Pistor
editors, Database Systems in Office, Engineer-
ing, and Scientific Environment, Springer Ver-
lag, New York (NJ), 1985.

[Newman 1983] W.M. Newman and T. Mott. Officetalk-
Zero: An Experimental Integrated Office Sys-
tem. In: P. Degano and E. Sandewall
editors, Integrated Interactive Computing
Systems, North-Holland Publishing Company
Stresa (Italy), 1982.

[Nievergelt 1979] J. Nievergelt and J. Weydert. Sites, Modes, and Trails: Telling the User of an Interactive System Where He Is, What He Can Do, And How To Get Places. Technical Report 28, Institute for Informatics, Swiss Federal Institute of Technology (ETH), Zurich (Switzerland), 1979.

[Nievergelt 1984] J Nievergelt et al. The GRID FILE: An Adaptable, Symmetric Multi-Key File Structure. ACM Transactions on Database Systems, 9(1), 1984.

[Orenstein 1986] J. Orenstein. Spatial Query Processing in an Object-Oriented Database System. ACM-SIGMOD, International Conference on Management of Data, 15(2), 1986.

[Shneiderman 1983] B. Shneiderman. Direct Manipulation: A Step Beyond Programming Languages. Computer, 16(8), 1983.

[Smith 1983] D.C. Smith et al. Designing the Star User Interface. In: P. Degano and E. Sandewall, editors, Integrated Interactive Computing Systems, North-Holland Publishing Company, 1983.

[Smith 1986] T. Smith et al. KBGIS-II: A Knowledge-Based Geographic Information System. Technical Report TRCS86-13, Dept. of Computer Science, University of California, Santa Barbara (CA), May 1986.

[Stonebraker 1976] M. Stonebraker et al. The Design and Implementation of INGRES. ACM Transactions on Database Systems, 1(3), September 1976.

[Tamminen 1982] M. Tamminen. Efficient Spatial Access to a Data Base. In: ACM-SIGMOD, Orlando (FL), 1982.

[Zloof 1977] M.M. Zloof. Query-by-Example: A Database Language. IBM Systems Journal 16(4), 1977.