MAPQUERY:

Data Base Query Language for Retrieval of Geometric Data and their Graphical Representation

André Frank

Institute of Geodesy and Photogrammetry
Swiss Federal Institute of Technology (ETH)
Zurich, Switzerland

Abstract

The storage of geographical data is a growing type of application within data processing.

Users want to retrieve parts of the data stored and to have them shown in graphical form. They need an easy and flexible query language to select the data needed and to describe the form of representation. The existing query languages do not suit this purpose without amendments. This paper presents a proposal for adaptation and gives some examples for query formulation. Moreover, it discusses the special problems of graphical input and output.

CR Categories and Subject Descriptors:

H.2.3 |Database Management|:
Languages - Query languages;

I.3.6 |Computer graphics|:
Methodology and Techniques -
Interaction techniques;

J.1 |Administrative Data Processing|:
Government

J.2 |Physical Sciences and Engineering|:
Earth and athmospheric sciences

General Terms: Human Factors, Languages

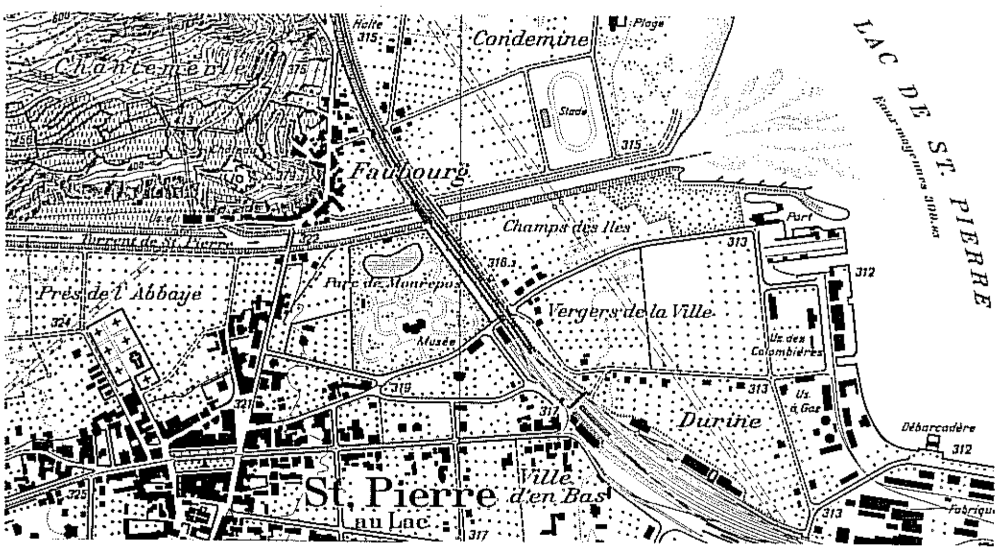
1. Introduction

Large amounts of data are stored in computer systems. Very often, graphical presentation is preferred; it seems that human beings understand pictorial information better than large tables of alphanumeric items.

An important problem in connection with the application of computer graphics is the form in which a casual user with little training can express his information needs. A query language adapted to the special requirements of graphical representation is apparently needed |12|.

This paper proposes a formal language for data retrieval and presentation. Formal, because a formal language seemed easier to implement in our environment lacking experience with natural language systems. It is, furthermore, uncertain whether it is worthwile to implement a natural language interface for this application. Although some working systems are reported |1|, |23|, |26|, it is not absolutely certain whether a natural language is appropriate as a query language |31|, |38|. As users are not conversing frequently about maps and their appearance, there is no fixed, widely used vocabulary on which to base a natural language query system |22|, |25|.

Figure 1



A number of substantially different types of data storage and graphical representation applications can be differentiated:

- retrieval of previously stored complete pictures usually in raster format (references may be found in |5|, |6|, |8|),
- representation in graphical form of data which is usually shown in tabular form (spatial data management system) |21|,
- retrieval and representation of data, usually presented in the form of geographic maps |15|, |16|, |29|.

This paper will treat the latter case only; it is, however, felt that part of the conclusions may easily be extended to the area of spatial data management system, too.

Based on a concrete simplified application, the needs of users for retrieval and representation of information are described. The well known SEQUEL language |2|, |3| is used as a starting point to develop MAPQUERY which is believed to be especially suited for retrieval of geometric data.

2. The Application

Land Surveyors traditionally gather data related to objects in space and represent these data on maps of different style, size and scale. These maps are used by a wide variety of users to many different purposes.

The gathering of geometric data is a costly process and closely related to the representation of data in the form of maps. The drawing of clear, easily understandable maps is a tedious and very expensive operation demanding highly qualified personnel (fig. 1).

In our society land has become a resource of utmost importance, requiring very careful management. All planning activities in connection with e.g. land use or environment protection need data about objects in space as they are traditionally shown on maps. But for different planning purposes different data are needed and often the form should be specially adapted. Traditional surveying techniques make it difficult to transfer information represented on one map into a new map of different style. The process requires expensive and time consuming manual work. More complicated - if not impossible - is very often to combine in one drawing data represented on different maps.

The possibilities of modern electronic data processing induced the idea to store land related data in a computer and to draw the desired maps on demand, exactly suited to the use intended.

3. Limitations of Available Specialised Systems

For different types of users - e.g. public utility companies - such systems have been developed to fulfil certain needs of these users - e.g. drawing of plans of the actual situation of pipes and lines in underground. Similar examples can be shown for planning applications. |15|, |27|, |41|, |47|.

In the past these systems have been based on storage and retrieval of the graphic features of the plans. The different map sheets are usually attributed to different files and in each file the lines and other graphical features of the map are stored. These systems do normally not provide a powerful interactive user query language. The selection of the objects on the screen is usually done by indicating the name of the map-sheet and then 'zoom' and 'pan' may be used to choose the desired window. It is obvious that such systems allow only for a very limited flexibility of adaptation to users needs, i.e. different query strategies, contents and representations.

But systems should be built to fulfil the needs of a wider group of users - a requirement that is certainly reasonable if we compare the costs of constructing such a system (which are very likely to be in the order of millions of dollars) to the costs of data gathering and maintenance (for a larger city a complete survey will cost tens of millions of dollars) |42|.

We call these systems Land Information Systems (LIS) because they are intended to store land related (i.e. spatially referred) data. Such Land Information Systems may not be based on stored graphical representations of objects but must store an internal (abstract) model of objects, from which model different graphical representations can be derived upon demand.

4. Data Base Management Systems

The theory concerning data base systems |13|, |46| and the available Data Base Management Systems are very valuable for implementing Land Information Systems.

The data model which has to be stored is to a certain extent similar to the data in the commercial environment where such systems have been succesfully applied.

In order to minimize the problems of updating stored information, data base theory states certain rules for splitting up information into independent parts (often called entities). Generally the goal is to minimize redundancy by storing one piece of information in one place only - for example, even if a person owns several houses, the address of the person is not stored as a data element in connection with each house but as an attribute of the entity "person". With the houses only a logical connection to the owner is stored; using this schema of storage it is evidently much simpler to update the owner's address.

5. Difference between Ordinary Commercial Data and Geo-Data

There is a great difference between ordinary commercial data and geometric data in a Land Information System. All data in a Land Information System are related to objects in space; they are, therefore, also related to a position in space. This is in contrast to commercial and administrative systems, in which data are hardly ever exploited in connection with position in space.

This relation to space and geometry is important to consider. Between the different data elements geometric and topological relations exist, e.g. a house is situated on a parcel, a parcel is adjacent to a street.

There are so many potential relationships that it is impossible to store all of them explicitly; but they can be stored implicitly in the form of coordinate geometry. However, the deduction of topological information from geometry may be quite expensive.

The second difference is in the form of access to the data. Commercial data are very often requested in queries yielding one or more selected data elements, presented in the form of a list.

As an example:

List all *employees* with *age higher than 55!*

which yields fig. 2 as an answer.

Name	Age
Ares, Jan	57
Myer, Don	60
Orlos, Pit	58
Pier, Mark	57

Figure 2

It is natural to represent objects in Land Information Systems geometrically, for which reason these systems are usually queried in order to get a map on the screen:

Show all *houses, streets, parcels* within *'Lohzelg' area!*

which yields fig. 3 as an answer.

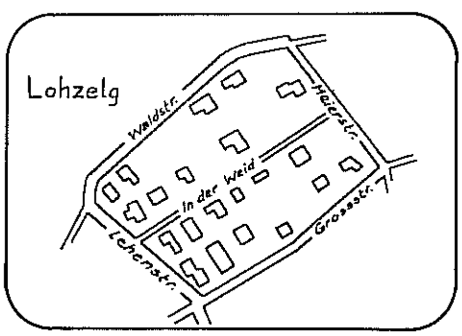


Figure 3

This typical query may be generalized into the form

show all <object-type-list>
within <window>!

where <window> specifies a subset of the two-dimensional space (range in X and Y) and is used to select a great many of data elements, which are shown as a map.

It is very important that this type of query can be answered fast. The storage structure should support fast access to data in spatial neighbourhood. This fast access is important not only for the queries, but also for fast geometrical and topological algorithms (e.g. in case a topological relation must be deduced from the co-ordinate geometry). Such a storage structure has been implemented at our institute |17|. It can be based on a commercially available Data Base Management System, i.e. DBMS-10 from DEC, a CODASYL network type, and should form the base for building a Land Information System.

6. A Query Language for Geo-Data

The query languages in commercial Data Base Management Systems are often based on relational calculus |14|; many are similar to SEQUEL|2|, |3|, which will serve as a background for the following discussion. It has been shown that these languages are 'relationally complete' i.e. all queries that can be formulated in first order predicate calculus may also be formulated in this language |10|. 'Completeness' is regarded as a minimal requirement for the expressive power of a query language and the need for extensions is generally felt |1|, |9|, |40|.

Expressing topological (e.g. 'a is within b') or geometrical (e.g. 'a is nearer to b than c') conditions in relational calculus is difficult if not impossible |38|. It will therefore be necessary to extend the language to include such specific functions, as e.g.

WITHIN: object x window → boolean

which is true, if the object or part of it extends into the window. WITHIN evokes therefore an operation like 'clipping' and cannot easily be expressed by boolean expression (c.f. fig. 4).

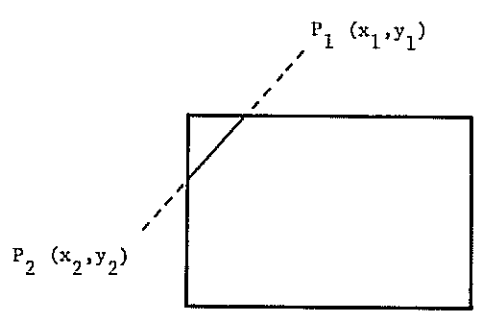


Figure 4

The special type of information the user expects makes other changes necessary. It is not simply a change in the form of representation of retrieved data from alphanumeric to graphical, but a far-reaching change in the form of man-machine communication.

The most important points to consider in this respect are:

- context needed (environment)
- selection through geometric or topological relationship (e.g. window)
- form of presentation
- graphical input.

6.1 Environment

If a user wishes a point to be represented, he will accept the output of the co-ordinate values of this point in alphanumeric form (fig. 5). But he will certainly not accept the point represented graphically on the otherwise empty screen (fig. 6) - he is simply unable to interprete the data.

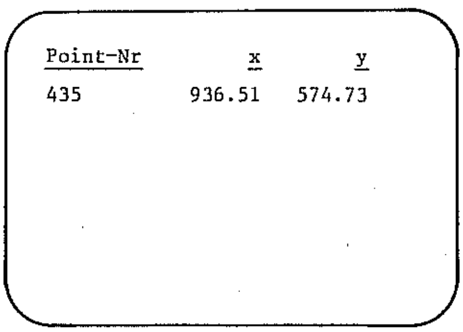


Figure 5

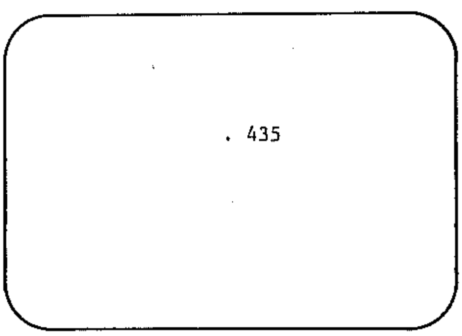


Figure 6

Therefore, in the graphical representation not only the object asked for but a suitable environment for interpretation must be presented. This is probably one of the reasons for the superiority of graphical communication.

'In general, personnel gather too little information prior to making a diagnostic decision. However, when more information is available, subjects are more efficient in selecting data items' |30|. 'This suggests that it may be wiser to provide the user with *more* information than he or she appears to ask for in answer to a query' |43|.

Two different aspects of the environment needed for interpretation have to be differentiated:

- the contents (e.g. houses and/or streets)
- the spatial extension (the window).

6.2 Window

In a query which should yield an answer in graphical form the user must set - implicitly or explicitly - a window |32| indicating the part of the world which is to be shown on his screen.

To set the window explicitly, the user must be given a command to indicate the limiting co-ordinate values (x.min, x.max, y.min, y.max). In many cases, this would be too troublesome for the user; he wishes to see a certain house of which he knows the postal address but not the co-ordinates.

It is the system's task to find out these co-ordinates! The user should be allowed to state a query like:

Show *house* with *address* = '23 Parkland Drive'

and expect the system to choose a suitable window to show the desired house with a reasonable part of the environment.

The window cannot be determined by the extreme coordinates of the object desired alone, but must be extended in order to include some of the surrounding objects. Some experiments show that users prefer the drawings not to be in an arbitrary scale, but are more at ease with some standard values. To decide whether a map is in the scale of 1:1000 or 1:500 is easy (based on e.g. width of streets) and unconsciously done. The user may then interpret the whole contents in accordance to his preexisting concept of a 1:500 scale map. This seems preferable to a scale adjusted to a maximum size on the screen, arriving at arbitrary scales, which make interpretation difficult.

First Example Query

The user desires a traditional map showing streets and street names, houses with numbers, parcel boundaries (fig. 7) within a window given by two corner points.

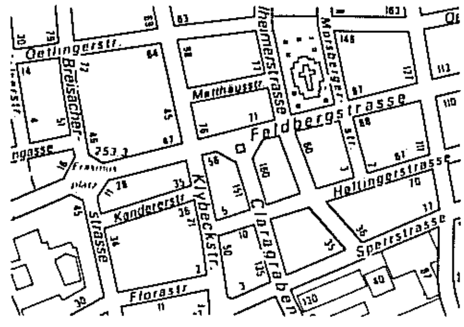


Figure 7

Although an abbreviation should be prepared for this query, we will discuss it here in detail. The concept of extensibility of a language to let the user define a query with the normal query language and then store this query and let him call it later by an abbreviation is certainly very useful |11|, |44|.

The data base provided for answering this query contains the following data-types (fig. 8).

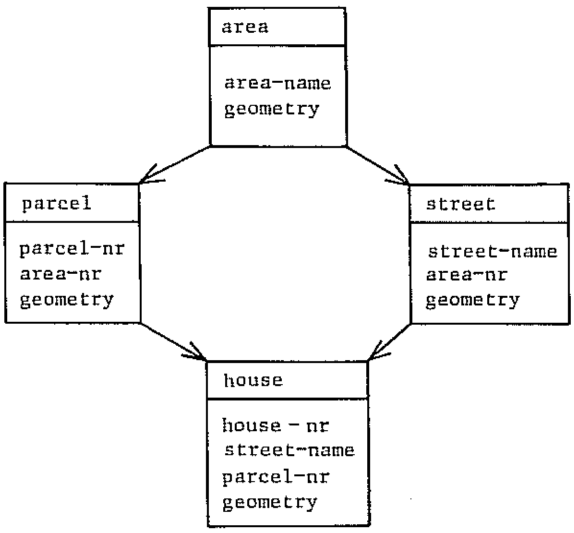


Figure 8

This schema naming the entities with their data fields and indicating possible connections by arrows is intended for ease of presentation only. The problem of designing optimal data description for geometric data is treated in |19|, |42|.

We can now express this first query in a SEQUEL like language:

```
DEFINE window (x.min, x.max, y.min, y.max)

SHOW    street-name, geometry
FROM    street
WHERE   geometry WITHIN window

SHOW    house-nr, geometry
FROM    house
WHERE   geometry WITHIN window

SHOW    geometry
FROM    parcel
WHERE   geometry WITHIN window
```

Certain liberties have been taken with SEQUEL to express this query but the result is not very promising for daily users. SEQUEL was developed for the selection of data through complex criteria - it seems less useful for the selection of several groups of data using always the same very simple criteria, i.e. the window.

It is certainly adequate - as it is usual in computer graphics languages |32| - to extend the effects of the WINDOW-statement from the place where it is given until it is explicitly recalled or overridden by a new WINDOW-statement. The WINDOW command sets, as understood by the user, a context which must be used to interpret his further commands |24|. Second, it seems superfluous to restrict each object selected by the condition that it extends into the window - this is from the user's point of view natural, otherwise it cannot be shown (and the user evidently does not care whether it is retrieved or not).

This would give:

```
IN WINDOW (x.min, x.max, y.min, y.max)

SHOW  street-name, geometry  FROM  street
SHOW  house-nr,    geometry  FROM  house
SHOW               geometry  FROM  parcel !
```

Second Example Query

The user desires a similar map as before but with a larger scale and featuring the house with address '7 Hill Drive' in the center (fig. 9).

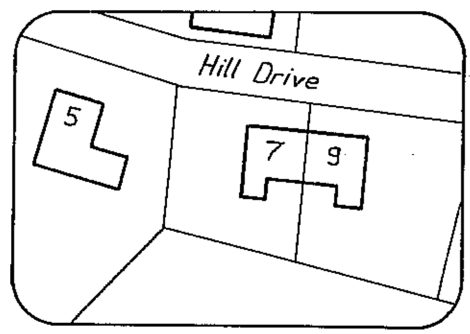


Figure 9

SEQUEL (and most other query languages) requires the user to write two different query statements: one for selecting the window and another for selecting the contents within the window.

A specialized language for retrieval of geometric data should offer a simpler formulation:

```
SHOW
IN WINDOW
THE  geometry  FROM  house
       WHERE street-name =  'Hill Drive' AND
                            house-nr = 7
ALL  geometry, house-nr     FROM  house
ALL  geometry, street-name  FROM  street
ALL  geometry,              FROM  parcel!
```

This query would effectively select the desired house twice, firstly in the 'THE' statement where the window is built, and secondly in the 'ALL...FROM house' phrase. This seems no real problem for CRT displays - for plotters it would be necessary to select first the window and check the output on a display and then ask for a new drawing with the previously set window:

```
SHOW
ALL  geometry, house-nr      FROM  house
ALL  geometry, street-name   FROM  street
ALL  geometry,               FROM  parcel!
```

The same construct allows also to build a map gradually, first selecting an entity which sets the window and then adding further data to the picture.

Third Example Query

The user needs the same type of map as before, but showing streets, houses and parcels for the area 'Talacher' (fig. 10).

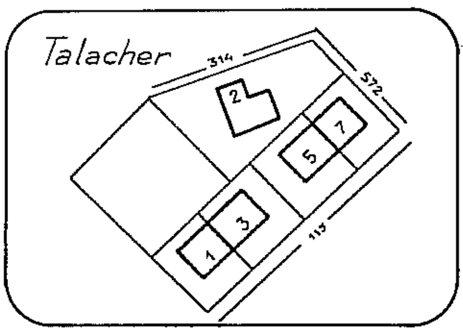


Figure 10

```
SHOW
IN WINDOW
THE geometry  FROM  area
WHERE area-name = 'TALACHER'
ALL geometry, street-nr FROM  street
    WHERE  area-nr  =
           SELECT  area-nr  FROM  area
           WHERE   area-name = 'TALACHER'
ALL geometry  FROM  parcel
    WHERE  area-nr  =
           SELECT  area-nr  FROM  area
           WHERE   area-name = 'TALACHER'
ALL geometry, house-nr  FROM  house
    WHERE  parcel-nr =
           SELECT parcel-nr FROM  parcel
           WHERE  area-nr =
                  SELECT  area-nr  FROM  area
                  WHERE   area-name = 'TALACHER'!
```

An easier solution is obviously needed:

The natural hierarchical structure of area containing streets and parcels, and parcels containing houses should be exploited.

The user starts with his naming of 'area' a context for the rest of the query. He can now ask to see the streets thereof (understood: the streets of this area), the parcels (understood: the parcels of this area) and the houses thereof (understood: from the parcels selected - the context has been moved). This context can be interpreted, given the connections between area, street and parcel and between parcel and house shown in fig. 8. To stress this context, it seems better to change the 'field FROM entity' phrase in 'entity (field)' |33|. This proposal exploits mainly ideas from |35|, |37| with different keywords and some minor differences in syntax.

```
SHOW
IN WINDOW
THE  area  (geometry)  WHERE  name  =  'TALACHER'
THEREOF  ALL  streets  (geometry, street-nr)
         ALL  parcels  (geometry)
              THEREOF  ALL  houses (geometry,
                                    house-nr)!
```

This is a unique description as long as there are not two different connections between two object-types. During the design of the user's view of the sorted data such problems can be catered to.

Fourth Example Query

The user needs a map showing the street named 'Hill Drive' and all the adjacent parcels.

In a schema in which a connection between street and parcel already exists (meaning the parcel which is formed by the street), we have in order to prepare this type of query to include a new object in the user's view: adjacent parcel to a street (fig. 11). With this solution we can dispense with including commands for selection based on topological relationship (c.f. |45|).

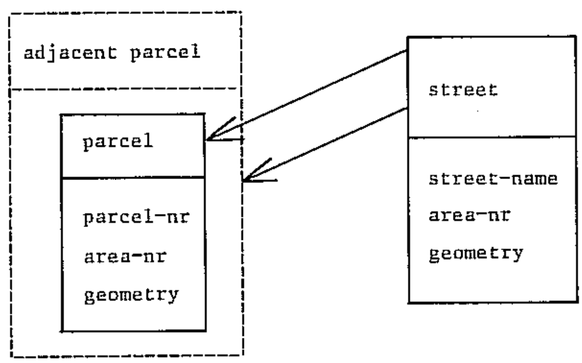


Figure 11

This allows the query

```
SHOW
IN WINDOW
THE street (geometry)  WHERE street-name =
                                'Hill Drive'
THEREOF ALL adjacent-parcels (geometry)!
```

7. Form of representation

For each application a basic list of conventional signs has to be prepared. This list should include signs with gradual differences to represent different classes of similar objects (fig. 12).

These conventional signs are implemented as procedures, using the routines of a general graphics package. Provisions for adding new signs should be made.

The query language must include a command to choose a certain sign for a class of objects.

We propose

```
SHOW ...        ALL  streets (geometry)
                     AS  street-sign-1!
```

house-sign-1   house-sign-2   house-sign-3

Figure 12

Further, the query language should allow to separate one group of objects into several classes depending on the values in a certain field.

An example how to graphically differentiate houses depending on their ages:

```
SHOW IN WINDOW THE area (geometry) WHERE
                      area-name = 'TALACHER'
THEREOF ALL streets (geometry)
        ALL houses  (geometry)
            WITH year-built <1850 AS Hous-sign 1
            WITH year-built >1850 AND year-built
                 >1900  AS Hous-sign-2
            WITH year-built >1900 AS Hous-sing-3!
```

There a very interesting problem appears: which objects can be represented by which conventional signs.

Traditionally, conventional signs in cartography are classified into points, lines and areas with distinct or open limits. Further research will show what types of signs can be used to represent which objects. Two different aspects must be considered.

- the possibility of interpretation of the result by the human user
- the interfacing between the data producing routines of the data base system and the routines for drawing.

8. Conventional Form and Contents of Output

In daily routine the user must be able to state in the shortest possible form that he wishes a certain representation. It is customary to see certain things in a map, as customary as to use certain signs to show the objects.

It is therefore necessary that with evoking a prestored map type the user not only selects the cartographic signs to be used for the different entities, but also a standard selection of contents.

In these cases, the formulated query rather describes the window to be chosen (e.g. by stating a uniquely selected object) or indicates some additional (preferably highlighted) information asked for.

In our proposition for a query language, after SHOW the name of the type of presentation needed can be included.

In this short form the query from the second example is only:

```
SHOW map-type-1 IN WINDOW
THE house (geometry) WHERE street-name = 'Hill Drive'
                           AND  house-nr = 7!
```

but the result shows also houses, parcels and streets, i.e. all the details needed for interpretation, because the query is automatically completed with the prestored selections.

9. Graphical Input

If some pictures are already drawn on the screen, it seems most natural to formulate a query with graphical input.

There is little difficulty to permit the selection of the key words of a query with a menu and a pointing device |36|.

More important is to be able to include the variable parts of a query. The last query would be simpler, replacing the description of the house by pointing on the screen:

```
SHOW  map-type-1   IN WINDOW
    THE house (geometry)   X !
```

where the 'X' indicates that the user will indicate the house desired by pointing on the screen (this is similar to 'Query by Pictorial Example' |4|).

In case where more than one object can be identified through pointing at one point (e.g. house or parcel), this requires a subtle interaction between the query analyzer and the PICK function. Depending on the syntactical context, the result must be the address of a house or the number of a parcel.

10. Related Work

In |5|, |8| an overview of pictorial and geographical data bases and their query languages has been given. Therefore, we can restrict our discussion to the main features of the present proposal.

The language proposed here is designed for 'skilled frequent user' |38|. It seems of prime importance that simple daily problems can be solved with short queries easy to formulate: nevertheless, extensions to specify more complex retrievals should be possible. We therefore opted for a formal language, permitting well chosen default values and did not consider systems like Query-by-Example |48| or derivates extended to graphical applications QPE |4|, IDAMS |34|.

Some systems (GEO-QUEL |20|, CADS |28|, GRAIN |7|) separate the selection of a map's contents from its display. This allows for an easy implementation of manipulation routines treating the retrieved map-data, but we preferred to hand a single step solution to the daily user. Nevertheless, such a splitting-up could easily be introduced in the design here proposed.

11. Conclusion

A query language for retrieval of geometric data and their graphical representation has to be adapted to the special needs of such systems. Traditional query languages for commercial systems cannot be used without changes.

The special problems are

- within the limits of the screen a multitude of objects have to be shown,
- window is selected either with coordinate values or 'reasonable' in respect to an object of interest,
- language must allow for constructs yielding several object types shown independently on the same screen,
- user should be able to evoke with one name a prestored style of representation,
- this style includes also a standard choice of object types,
- differentiation of object classes based on an attribute and their differentiated representation must be possible,
- graphical input must be acceptable wherever the user prefers this form.

A proposal for such a query language has been described. This proposal features:

- automatic adjustment of a window including the object of interest,
- use of context during selection of objects in order to shorten query specifications,
- including selection of cartographic signs for representation,
- recall of standard selection and standard representation.

It is implemented in a restricted fashion in a prototype of a Land Information System at ETH |18|.

Acknowledgments

I thank Prof. R. Conzett and Prof. C.A. Zehnder (Institut für Informatik) and their assistants for many fruitful discussions. Special thanks go to Beat Sievers and Bruno Späni who during years of practical work came to know the needs of the users and looked to the appropriateness of this query language.

Comments on an earlier version of this paper by Lothar Gründig (Heidelberg) and Markku Tamminen (Helsinki) helped much to clarify the ideas.

The reviewers' comments contributed much to the readability of this paper.

The implementation is done on the DEC-10 computer of the 'Zentrum für interaktives Rechnen' and I am grateful for the permission to use these computer facilities.

References

1. Blaser, Albrecht, Lehmann, Hubert: Abfragesprachen in Datenbanken, in: IBM Nachrichten 30 (1980) Heft 251.
2. Chamberlin, D.D., Boyce, R.F.: SEQUEL: A Structured English Query Language, in: Proc. of 1974 ACM SIGFIDET Workshop on Data Description, Access and Control, Ann Arbor Michigan, 1974, p. 713 - 775.
3. Chamberlin, D.D. et al.: Sequel 2: A unified approach to data definition, manipulation and control, in: IBM J. Res. Develop. 20 (1976) p. 560 - 575.
4. Chang, N.S., Fu, K.S.: A Query Language for Relational Image Database Systems, Proc.IEEE Workshop Picture Data Description and Management, Aug. 1980, pp. 67 - 73.
5. Chang, N.S., Fu, K.S.: Picture Query Languages for Pictorial Data-Base Systems, Computer Nov. 1981, p. 23
6. Chang, S.K., Fu, K.S. (Ed.): Pictorial Information Systems, Springer-Verlag 1980
7. Chang, S.K., Lin, B.S., Walser, R.: A Generalized Zooming Technique for Pictorial Data Base Systems, AFIPS Conf. Proc., 1979 NCC, Vol. 48, pp. 147 - 156.
8. Chang, S.K, Kunii, T.: Pictorial Data-Base Systems, in: Computer Nov. 1981, p. 13.
9. Childs, D.L.: Extended Set Theory. A general model for very large, distributed, backend information systems, in: 3rd Internat. Conf. on Very Large Data Bases, Tokyo 1977.
10. Codd, E.F.: Relational Compleneness of data base sublanguages, Courant Computer Science Symposium 6, "Data Base Systems", R. Rustin (Ed.) Prentice Hall, 1972, IBM RJ 987.
11. Codd, E.F.: Seven Steps to Rendezvous with the casual user, in "Data Base Management" (Ed.: Klimbie/Koffeman) North Holland, Amsterdam 1974.
12. Conzett, R.: Landinformationssystem, Vortrag an Tag der offenen Tür, 12. April 1980, Institut für Geodäsie und Photogrammetrie, ETH Zürich, 1981
13. Date, C.J.: An introduction to database systems, Addison Wesley, Reading 1979, 2nd ed.
14. Dieckmann, E.M.: Three relational DBMS (Oracle, Ingres, SQL/DS), Datamation Sept. 1981.
15. Dutton, G. (Ed): First international advanced study symposium on topological data structures for geographic information systems, Harvard Papers on Geographic Information Systems, Harvard University, Cambridge, March 1978.
16. Eichhorn, G.: Land Information Systems. Symposium of the Fédération Internationale des Géomètres, October 16 - 25, 1978, Darmstadt

17. Frank, A.: Applications of DBMS to Land Information Systems. 7th Internat. Conf. on Very Large Data Bases, Cannes (France), 1981, p. 448.

18. Frank, A.: MAPQUERY: Design and implementation, Problems of realization of Land Information Systems, Part 2, Report 56, Institut für Geodäsie und Photogrammetrie, Swiss Federal Institute of Technology, Zurich, Switzerland.

19. Frank, A.: Data Structure and Data Description for Geo Data Systems, Problems of realization of Land Information Systems, Part 3, Report 57, Institut für Geodäsie und Photogrammetrie, Swiss Federal Institute of Technology, Zurich, Switzerland.

20. Go, A., Stonebraker, M., Williams, C.: An approach to implementing a Geo-Data System. Electronics Research Laboratory, College of Engineering, University of California, Berkeley, Meno ERL-M 529, 25 June 1975.

21. Herot, Christofer; F., Carling, Richard; Friedel, Mark; Kramlich, David: A Prototype Spatial Data Management System. Siggraph '80, ACM Computer Graphics, Vol. 14, No. 3, p.63, July 1980.

22. Hooper, K.: Experimental Mapping. The perceptual representation of environments. University of California, Santa Cruz, Technical Report.

23. Johnson, J.: Intellect on Demand: INTELLECT, Datamation Nov. 1981, p.73.

24. Jones, P.F.: Four principles of man-computer dialogue, in: Computer Aided Design Journal, Vol. 10, No. 3, May 1978.

25. Krause, J., Lehmann, H.: User Speciality Language. A natural language based on information system and its evaluation. Int. Conf.on Literary and Linguistic Computing, Bonn, Dec. 1979.

26. Krause, J.: Natural Language Access to Information Systems. An Evaluation Study of its Acceptance by End Users, in: Inform. Systems, Vol. 5, No. 4, May 1980, pp 297-318.

27. M & S: The use of computer graphics for utilities mapping and records, Report 78-090 A, M & S Computing Inc. 1978.

28. Mantey, P.E., Carlson, E.D.: Integrated Geographic Data Bases: The GADS Experience, in: Data Base Techniques for Pictorial Applications, ed. by A. Blaser, Springer-Verlang, 1979, pp. 173 - 198.

29. Meier, A., Zehnder, C.A.: Flächenmodell-Register wichtiger geographischer Datensammlungen der Schweiz. Institut für Informatik ETH Zürich, Bericht Nr. 39, 1980.

30. Meister, D.: Behavioral foundations of system development. New York: Wiley, 1976.

31. Montgomery, C.A.: Is Natural Language an Unnatural Query Language, Proceedings of the ACM Annual Conference, New York, 1972, p.1075.

32. Newman, W.M., Sproull, R.F.: Principles of Interactive Computer Graphics, 2nd ed. Mc Graw-Hill 1979.

33. Reisner, P.: Human factors studies of Database Query Languages, a survey and assessment, ACM Surveys Vol. 13, No. 1, March 1981, p.13.

34. Schmutz, H.: The Integrated Data Analysis and Management System for Pictorial Applications, in Data Base Techniques for Pictorial Applications, ed. by A. Blaser, Springer-Verlag, 1979, pp. 475 - 493.

35. Senko, M.E.: Data description language in the concept of multilevel structured description: DIAM II with FORAL. Douqué BCM. Nijssen GM. Ed., Data Base Description 1975.

36. Senko, M.E.: FORAL LP: design and implementation, VLDB 1978, Berlin.

37. Senko, M.E.: A query-maintenance language for the data independent accessing model II, in: Inform. Systems, Vol. 5, 1980.

38. Shneiderman, B.: Improving the human factors aspects of database transactions. ACM Transaction on Database Systems, Vol. 3, No. 4, December 1978

39. Stonebraker, M.; Wong, E.; Kreps, P.; Held, G.: The Design and Implementation of INGRES, ACM Transaction on Database Systems, Vol. 1, No. 3, September 1976, p. 189.

40. Stonebraker M.; Rowe, L.A.: Observations on Data Manipulation Languages and their Embedding in General Purpose Programming Languages. Proceedings 3rd Int. Conf. on Very Large Data Bases, IEEE, New York, 1977, p. 128.

41. Synercom: Synercom Information and Mapping System, User manual. Synercom Technology, Inc., 1980.

42. Tamminen, M.: Management of spatially referenced data. Conceptual study of system requirements and structure. Report, Helsinki University of Technology, Finnland.

43. Thomas, John C.: Psychological Issues in Data Base Management. Proceedings 3rd Int. Conf. on Very Large Data Bases, IEEE, New York, 1977, p. 169.

44. Thompson F.B.; Thompson B.H.: Practical natural language processing. The REL system as prototype, in: Advances in computers, Vol. 13, New York 1975.

45. Tsurutani, T.; Kasakawa, Y; Naniwada, N.: ATLAS: A geographic data base system, in: ACM Computer Graphics, Vol. 14, No. 3, July 1980. SIGGRAPH 80, Conf. Proceedings.

46. Ullman, J.D.: Principles of Data Base Systems. Pitman London 1980.

47. Wild: Wildmap, Interactive Photogrammetric Data-Base and Mapping System. Operators Procedure Manual, Wild (Heerbrugg) 1980.

48. Zloff, M.: Query by Example, AFIPS Conference Proceedings, 1975 National Computer Conference, Vol. 44, AFIPS Press, 1975, p. 431.