

SUPPORTING MULTIPLE REPRESENTATIONS WITH SPATIAL DATABASES VIEWS MANAGEMENT AND THE CONCEPT OF « VUEL ».

Yvan Bédard and Eveline Bernier
Centre de recherche en géomatique (CRG)
Université Laval, Qc, Canada
Yvan.bedard@scg.ulaval.ca
Eveline.bernier@scg.ulaval.ca

KEYWORDS

GIS, VUEL, multiple representations, multidimensional database, SOLAP, spatial database view.

ABSTRACT

Fulfilling the needs of spatial database users may require the capability to describe a same reality according to different points of view as well as different abstraction levels. This is the case for example of web-based systems that support "maps-on-demand" (MOD) and systems built for the interactive exploration of spatial data (SOLAP: spatial on-line analytical processing). Such applications require a flexible database view engine that supports simultaneously geometric multiplicity, semantic multiplicity and graphical multiplicity. This paper presents the requirements specific to MOD and SOLAP with regards to multiple representations. In particular, fundamental concepts about geometric, semantic and graphical multiplicities are defined. Then, we introduce an innovative concept aimed at meeting these requirements: the VUEL (View Element). We believe this new solution is a step towards very flexible systems that can support different views involving different multiplicities and different levels of abstraction.

1. MAP-ON-DEMAND (MOD)

On-demand mapping is a new trend in cartography where the maps are built upon a set of constraints defined by the user. As opposed to one might think, this doesn't necessarily imply that the map should be created in real-time. In fact, the map can be built on-the-fly but can also be created in a matter of few hours and even of few days. The main characteristic is that the user specifies *when* and *upon which specifications* the map should be created [Weibel *et al.*, 2002]. The resulting map can thus be seen as an *ad hoc* representation of a given dataset.

When such capabilities are included in Web mapping applications, time becomes an important factor that has to be taken into account; on the Web, maps must be displayed as fast as possible. Current Web mapping applications offering MOD capabilities are rather limited, offering only some options to the users. Usually, they can select the scale (among a pre-existing coarse set) and the themes to be displayed. The user has no regards concerning the geometry of the displayed object or even its graphical characteristics. In order to better fulfill the needs of Web

users, we need to increase the richness of these capabilities. For instance, the user could have the possibility to choose 1) the objects to be displayed, 2) the granularity of their geometries, 3) the graphic semiology to be used to display the selected objects, 4) the level of semantic defining the objects, and so on.

2. SOLAP

Spatial On-Line Analytical Processing (SOLAP) is a new category of tools that aimed at supporting the exploration and analysis of spatial data in a very intuitive way. Bédard (1997) defines this type of tools as a “visual platform built especially to support rapid and easy spatio-temporal analysis and exploration of data following a multidimensional approach comprised of aggregation levels available in cartographic displays as well as in tabular and diagram displays”. The main characteristic of this kind of tools is that the user doesn’t have to know *a priori* the underlying database structure (e.g. table names, relations) as well as he doesn’t have to build complex queries in order to see what he wants (e.g. SQL queries). The whole navigation can be done exclusively by some mouse clicks. Consequently, we often refers to this type of tools as a “keyboard-less GIS”.

The multidimensional structure of SOLAP tools rely on some fundamental concepts such as “dimension”, “measure” and “fact”

that are the basis of multidimensional databases [HAN 98; THO 99; RIV 01]. *Dimensions* are the main analysis themes of an organization while *measures* are the attributes being analyzed against the different dimensions [RIV 01]. For example, in a sale application, “Product”, “Region”, “Store” and “Time” can be defined as dimensions while “Sales”, “Profit” and “Number of customers” can represent

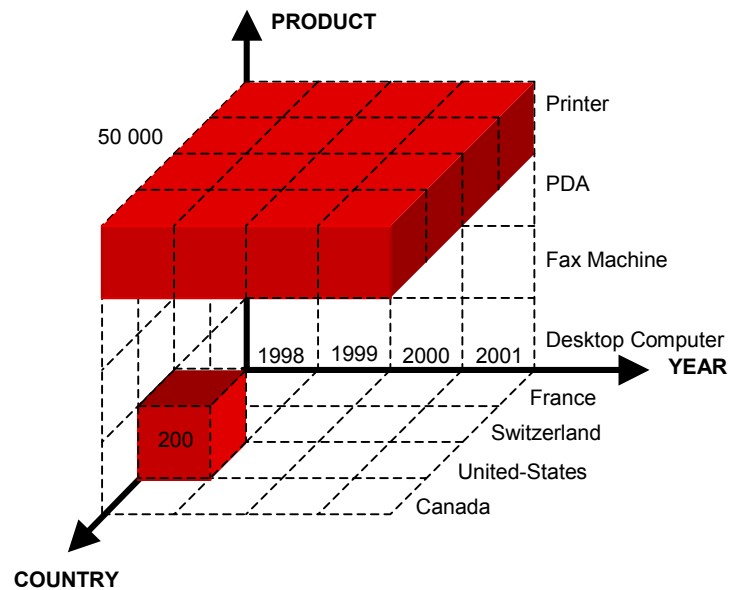


Figure 1 SOLAP Cube

measures. The latter can thus be seen as the dependant variables while dimensions are the independent variables, as the value of a measure, called "fact", is the result of the intersection of the dimensions [RIV 01]. Each dimension contains *members* that are organized in different hierarchical levels. These levels differ by their *granularity*, going from coarse to fine. Coarse

levels contain aggregated data while fine levels contain detailed data. For instance, the "Time" dimension could contain three levels, i.e. "Year", "Month" and "Day" where "1998", "March" and "26" could be members of these levels. The fact is then a measure value resulting from the intersection of at least two members that are part of different dimensions, thus the term "multidimensional". Finally, the grouping of several dimensions forms what is called a "hypercube" (given that usually, there are more than 3 dimensions). The figure above shows a cube formed by three dimensions :Country, Product and Year. Lets say that the measure we want to analyze is the number of products sold. The small lower cube indicates that in the United-States, they sold 200 desktop computers in 1998 while the upper cube shows an aggregation of the printers sold (50 000) for these four countries from 1998 till 2001. The values "200" and "50 000" are the facts resulting from these simple and aggregated intersections.

In a SOLAP tool, we interactively explore the database with operators such as drill-down, drill-up, drill-across and swap. The two first operators are used to navigate among the different levels of a given hierarchy, i.e. going from coarse levels to finer ones and vice versa. For example, in our TIME dimension, we can drill-down on the "Year" level to see the data by month and drill-up to return to the "Year" level. The drill-across operator allows a user to see another dimension at the same level of details while the swap is used to change one of the dimensions that are being analyzed.

Current SOLAP applications have only one geometric representation for each member of a spatial dimension. So, these members are meant to be visualized at a given scale or for a restricted range of scales. In fact, a significant reduction in the displayed scale, without any generalization, results in an unreadable representation. Inversely, a significant enlargement of the displayed scale does not convey the level of details necessary for the analysis being performed. Thus it appeared necessary for a user to zoom in or zoom out on a particular view to see the same members with more or less geometric details. Furthermore, when activated by the user, the semantic associated to the object could also change in synchronization with the level of details of the geometry providing a general description with a simplified geometry and a specialized description with a detailed geometry.

By offering the possibility to represent the spatial and semantic data according to different abstraction levels of for different needs, SOLAP tools and Web mapping applications with MOD capabilities would be more flexible and would have the ability to better fulfill the various needs of users. However, both kinds of applications require fast answer times as they are interactive applications. Given this constraint and the actual state-of-the-art of the on-the-fly generalization,

we believe that such systems must rely on a database structure supporting multiple representations.

3. FUNDAMENTAL CONCEPTS ABOUT MULTIPLICITIES

In general, when speaking about multiple representations, one refers to the geometric aspect of spatial data. A multiple representations database (MRDB) may thus store a same object according to different geometries. Nevertheless, we believe this definition must be enlarged to also take into account the semantic as well as the graphical aspects. Accordingly, there are three types of multiplicity that can be identified [Martel, 1999].

First, the well known **geometric multiplicity** refers to an object that can be represented with different geometries. This is normally the case when the same object can be displayed at different scales. For example, a building would be represented with a detailed polygon at a large scale and with a simplified polygon or even a point at a smaller scale (figure 2).

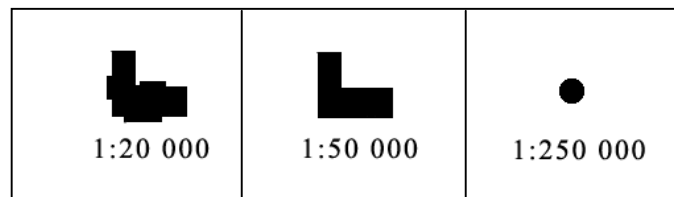


Figure 2 *Multi-scale geometric multiplicity*

However, there is another need to store different geometries for a same object. Nowadays, organizations tend to integrate all their data (coming from disparate, heterogeneous systems) in a data warehouse. This data warehouse serves many different applications and each of them has its own needs regarding how an object has to be displayed. For example, some applications need to display houses according to their roofing while other applications might display them according to their foundations (figure 3). We refer to this situation as a “single-scale geometric multiplicity”.

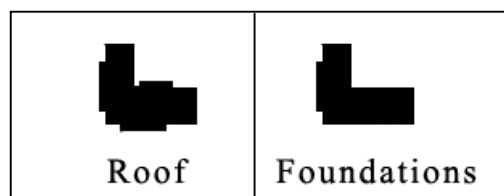


Figure 3 *Single-scale geometric multiplicity*

There also is the **semantic multiplicity** where the same cartographic element can be associated with different definitions, representing different concepts for different applications of a same data warehouse. These semantics can differ or not according to the scale. For example, a cartographic element can be defined as a house, a building or a construction (i.e. for a same dimension, three levels of semantic abstraction) depending on the view of the database. It could also be defined as a commerce, a residence or an industry (i.e a same level of semantic but different dimensions). As these definitions can vary depending on the application domain. Given that each application domain has its own ontology, a same cartographic element could have different meanings. Furthermore, one can mix semantic multiplicity, with geometric multiplicity as shown in figure 4.

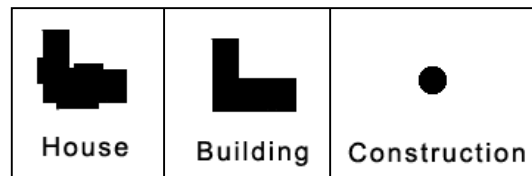


Figure 4 *Multiscale semantic multiplicity*

Finally, the **graphic multiplicity** occurs when a same cartographic element is represented by different visual variables following different semiology rules. As for the two other kinds of multiplicity, this one can be associated to a change in the display scale. For example, the symbols used to represent certain types of elements can be different from one scale to another, or from one application domain to another (figure 5)

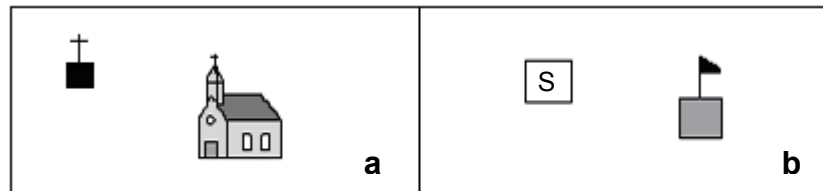


Figure 5 *Graphic multiplicity a) multi-scale b) single-scale*

The three types of multiplicity are necessary to fully harness the power of spatial data warehouses and to fulfill the needs of SOLAP and MOD users. As far as we know, no software solution has been developed insofar to support these multiplicities all at once. This is why we developed the VUEL concept, i.e. a new way to manage multiple views of spatial database.

4. THE VUEL CONCEPT

In order to support these three kinds of multiplicity in spatial databases, we've developed a new concept called VUEL, for View Element. A vuel is defined as the basic construct of a spatial database view, as the pixel (picture element) is the basic element of an image. This structure is well suited for SOLAP or Web mapping applications offering MOD capabilities that require fast answer times because vuels are organized in a multidimensional data structure where most data are stored explicitly and aggregation/generalization algorithms used only when possible. In this following section, we first present the theoretical and fundamental notions under this concept and then, we discuss about its implementation in a SOLAP context.

Inspired by the view mechanism in spatial databases [Claramunt & Mainguenaud], a vuel represents any element that can appear on a spatial database view [Bédard et al., 2000]. The vuel is in fact, a unique combination of a visible element (geometry and graphic semiology) with a given semantic [Bernier, 2002]. Accordingly, the vuel is not the semantic object in itself (e.g. house) but what we can see in a view, for example a striped blue polygon with house attributes. The same object of the reality can also be represented by a red point in another view or he can becomes spotted green and defined as a building with different attributes and so on (figure 6).

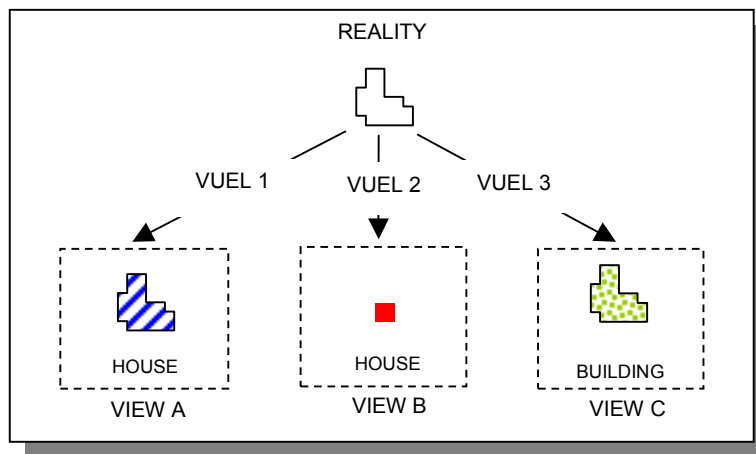


Figure 6 *Three vuels used to define or represent a same real world element*

In a SOLAP application, the data are presented to the user according to several display modes including cartographic representations, tables and statistical diagrams (e.g. pie chart). In that particular context, a vuel can then correspond to several concepts such as a cell of a table, a specific piece of a pie chart, a bar of an histogram, a pixel of a picture, etc. It can even correspond to some artifacts such as elements in cartographic or chart legends. Explaining the details of this characteristics goes beyond the goal of this paper, but mentioning that the concept extends to any type of database view adds to the interest of using vuels for maps.

4.1. THE VUEL MODEL

In the multidimensional database language, this model, made with UML notation, presents the VUEL as a fact table linking three dimensions: Geometric, Semantic and Graphic. This fact table is composed of all the ID that intersect from each dimension. In our case, each occurrence of that fact table thus represent a unique combination of a geometry, a semantic and a graphic semiology. A simplified model is presented in figure 7.

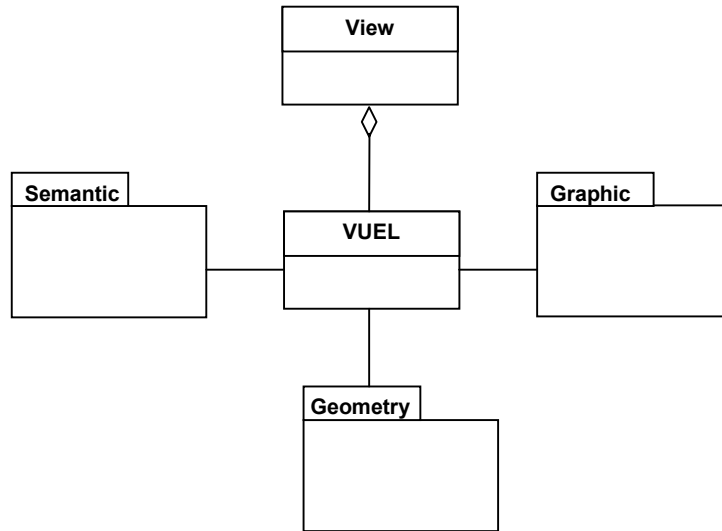


Figure 7 VUEL simplified model (UML notation)

In this vuel global model, we see that a vuel is a fact, i.e. a unique combination of a geometry, a semantic and a graphic semiology. This model also illustrates that a view is an aggregate of vuels. Different views are different aggregations of vuels and different vuels are different combinations of semantic, geometry and semiology.

THE GEOMETRY PACKAGE

First, to be visualized, each vuel is composed of a geometry (figure 8). This geometry can correspond to a cartographic element, a matrix element, a statistical diagram element, a picture element (pixel) or the element of any type of representation. In the case of cartographic elements, they can be primitives (eg. point, line or polygon) or artifacts such as those found in legends. Some SOLAP operations are also included in the Geometry class. These allow a user to navigate in the database to see the multiple geometries that are associated to related or same semantics for example. The spatial drill-down can be used to see the same semantic element with a more detailed geometry as opposed to the spatial drill-up that can serve to visualize a more simplified geometry. The spatial drill-across operation can be used to navigate between the geometries

associated to a same semantic element at the same level of granularity (single-scale geometric multiplicity). The recursive relation of generalization indicates that a geometric element can be a generalization of another geometric element and vice versa, facilitating the spatial drill-up and spatial drill-down.

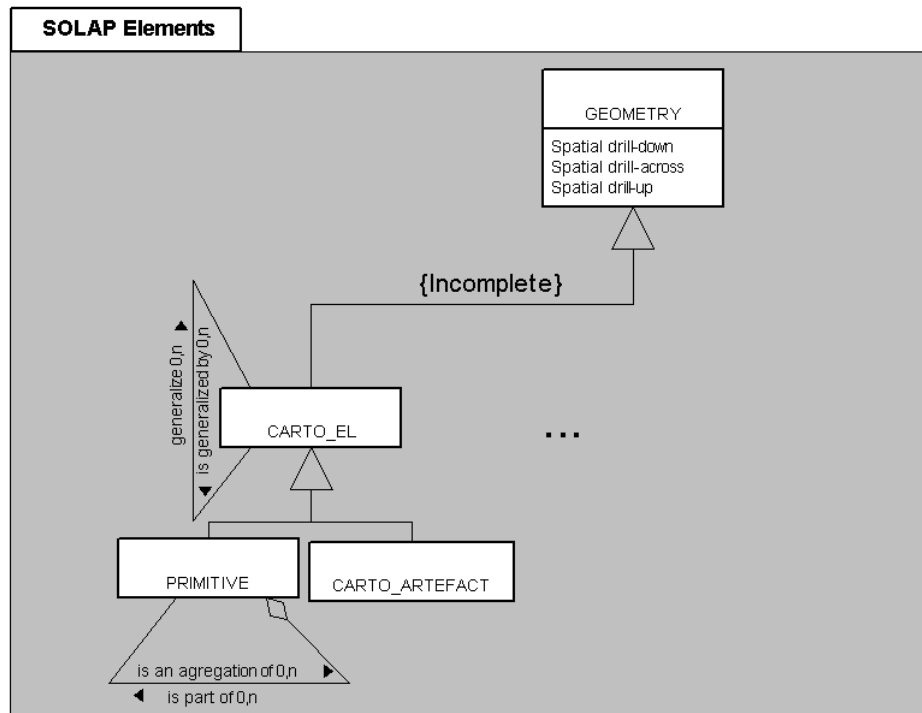


Figure 8 *The geometric package where only the cartographic element is detailed*

THE SEMANTIC PACKAGE

Secondly, in order to define an element, each view is composed of a semantic. As indicated on the conceptual model, the semantic is formed by a class name, that can be associated to some attributes which may have a domain of value (figure 9). The combination class-attribute-domain may change among views, each combination being a unique semantics. Accordingly, the semantic class is a second fact table that is used to make explicit all possible combinations of class-attribute-domain semantics. Doing so allows us to have very flexible views. For example, for one view, the class "House" could be associated to the attribute "market value" for a give polygon while in another view, the same polygon can be used for another semantic meaning, e.g. "building + use = residential" or "house + number of residents".

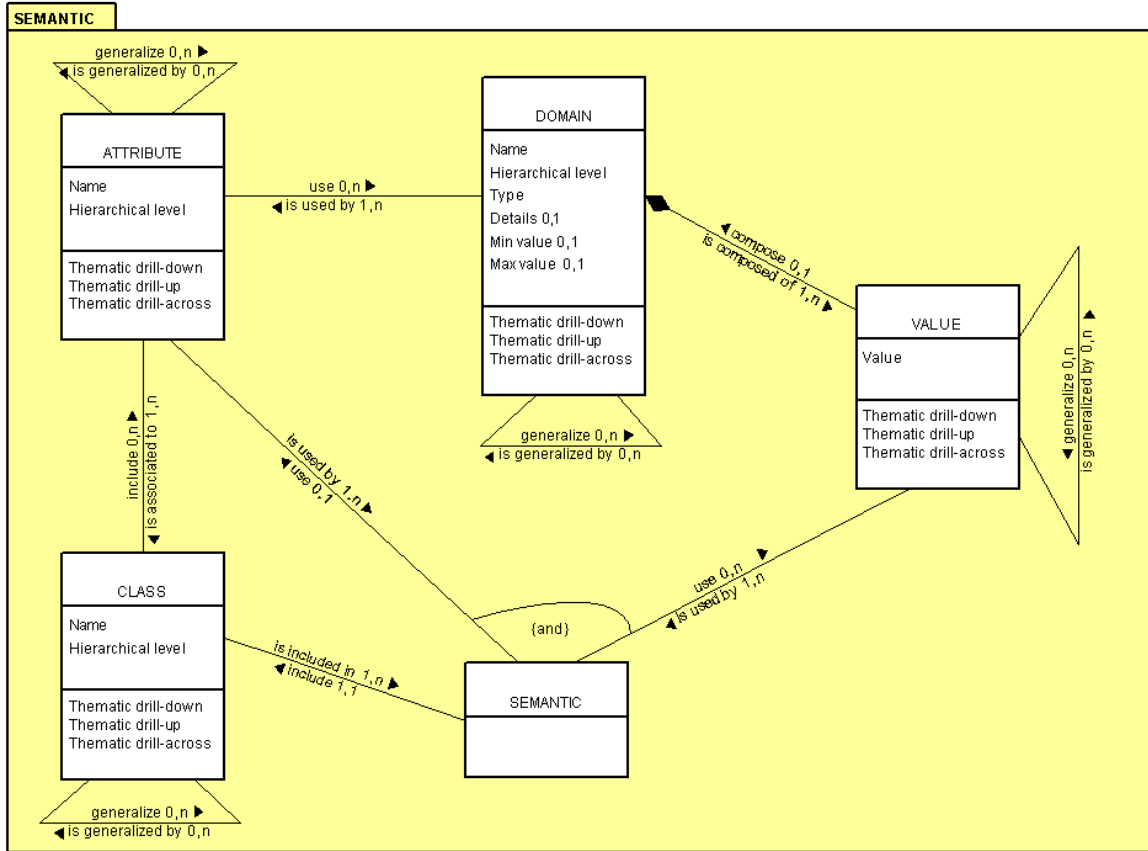


Figure 2 The semantic package

As for the geometric class, some SOLAP operations are defined for the semantic. For example, we can drill-down on the class that is associated to a cartographic element to have a more detailed description of that element. The same thing is possible for the attributes, the domains of values as well as for the values in themselves. These SOLAP operations are facilitated by recursive relations of generalization. For instance, the one associated to the "Class" implies that a class can be the generalization of another one. This allows for the management of class hierarchies according to the abstraction levels of a dimension. Value hierarchies associated to a domain that represent different abstraction levels can also be supported by the recursive relations associated to "Value". For instance, the attribute values "high-density residential, medium-density residential and low-density residential" used in one view can be replaced by the value "residential" at another more general view using the same polygons. These recursive relations are thus used to link, in a hierarchical way, these different values.

THE GRAPHIC PACKAGE

Finally, a graphical semiology is also linked to each vuel (figure 10). This semiology is determined by the visual variables that are : the color, the shape, the size, the value, the orientation and the texture in the case of cartographic elements. This is particularly useful for data warehouses that serve many applications where each of them has specific rules governing the graphic properties to be used to display certain types of objects. SOLAP operators are used to see different granularities of a same graphic representation

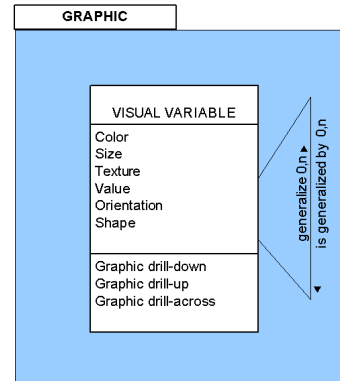


Figure 10 *Graphic package*

(graphic drill-down and drill-up) and different graphic representations for a same cartographic element (graphic drill-across). For example, a first view can show houses in blue while another view can show houses with different hints of blue according to a specific attribute (dark blue for houses where the market value is above 125 000\$, mid blue for the one that have a value market between 75 000\$ and 125 000\$, and finally, light blue for houses that have a market value under 75 000\$). The use of these operators are once again facilitated by a recursive relation of generalization.

THE VIEW CLASS

Finally, views are defined as an aggregate of vuel (figure 11). The SOLAP operators that we've seen so far are used to navigate between the different multiplicities associated to only one element. However, it is also possible to apply one of these operators for an entire view. For instance, one could make a spatial drill-down on a view to see all the cartographic elements with their detailed geometry (when possible). Additionally, as the user navigates from one database view to the next, it may become very useful to keep a trace of the database navigation operations in order to facilitate backward navigation. This is especially needed for the SOLAP applications used for geographic knowledge discovery. We thus defined a recursive relation to do so.

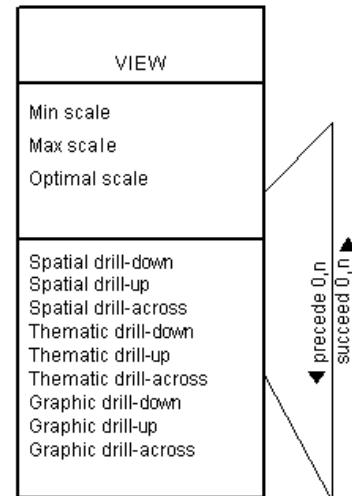


Figure 11 *The View class*

Finally, according to this conceptual model, the vuel becomes a way to record all types of multiplicities and the model itself becomes the structure of a *spatial database view engine* supporting multiple representations.

4.2. THE IMPLEMENTATION

In order to test these theoretical concepts, we've developed a SOLAP prototype based on the VUEL model and using real data coming from the Province of Quebec Topographic Database. This prototype allows the generation of alternative views of cartographic type. Because of the limited period of time to develop this prototype, the other forms of display (tabular and statistical diagram) have not been implemented.

For SOLAP applications, three types of architecture can be used : ROLAP, MOLAP and HOLAP. The first one, relational OLAP, relies on a relational database, the second, multidimensional OLAP, uses a multidimensional database (namely a "hypercube") while the third one, hybrid OLAP, is the combination of the two first architectures where the detailed data are stored in a relational database and the aggregated data are stored in a multidimensional database. Defining in more details these architectures goes beyond the aim of this paper, the reader is thus invited to consult [Rivest *et al.*, 2001; Thomsen *et al.*, 1999] for more details. For our prototype, we exploited the ROLAP architecture. When using a ROLAP architecture for analysis use (as opposed to transactional use), we have to denormalize our data structure. Notably, this will decrease the number of tables to link for a query, thus increasing the performance of data queries. To do so, different implementation models can be used (star schema, snowflake schema, etc.). These models simulate a multidimensional implementation. For our prototype, we used the snowflake schema (figure 12)

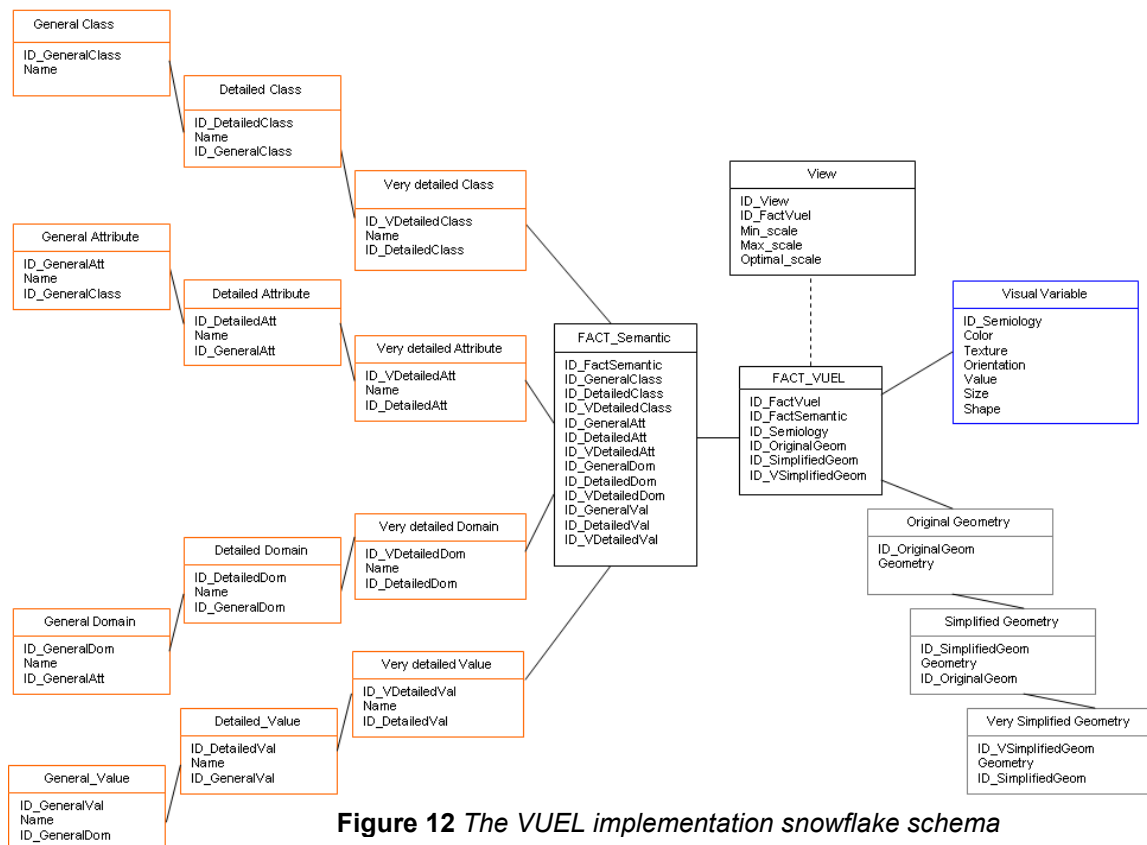


Figure 12 The VUEL implementation snowflake schema

As it is shown on the above figure and conformed to what we said earlier, each occurrence of the VUEL fact table represents a unique combination of semantic, graphic and geometric representation.

For our implementation, all the data have been stored in a Microsoft Access database, using the snowflake schema presented above (figure 12). The geometry of the objects has been included in the database via GeoMedia (Intergraph) which allows the inclusion of geometric tables in an Access database. We used DynaGen (Intergraph) to generalize the geometry of the objects in order to obtain three levels of geometric granularity. Furthermore, all the possible combinations (geometric, semantic and graphic) had to be determined at the beginning.

4.3. OVERVIEW OF THE SOLAP CLIENT

The SOLAP client, which allows a user to explore the data warehouse, has been developed with the Visual Basic language and uses GeoMedia object libraries for the cartographic aspect. The data are transparently reached by SQL (*Structured Query Language*) queries that are invisible to the user. This interface presents two parts; the descriptive part and the cartographic part (figure 14). First of all, the descriptive part allows a user to select the objects that he wants to visualize. This selection is primarily done by choosing the object classes (the upper part of the semantic tab). These object classes are presented to the user according to different levels of abstraction (visible via an expansive(+)/retractable (-) tree). It is thus possible to select general (hydrographic, roads, buildings, etc.) or specific classes (rivers, highways, school, etc.). Then the attributes corresponding to the selected class are shown in the middle part of the semantic tab, according to different abstraction levels. The user can then refine its selection by specifying the value of certain attributes. If these attributes don't have a domain of values, the user is invited to enter a value. In the opposite case, the possible domains of values corresponding to the selected attribute are displayed in the lower part of the semantic tab, once again, at different levels of abstraction.

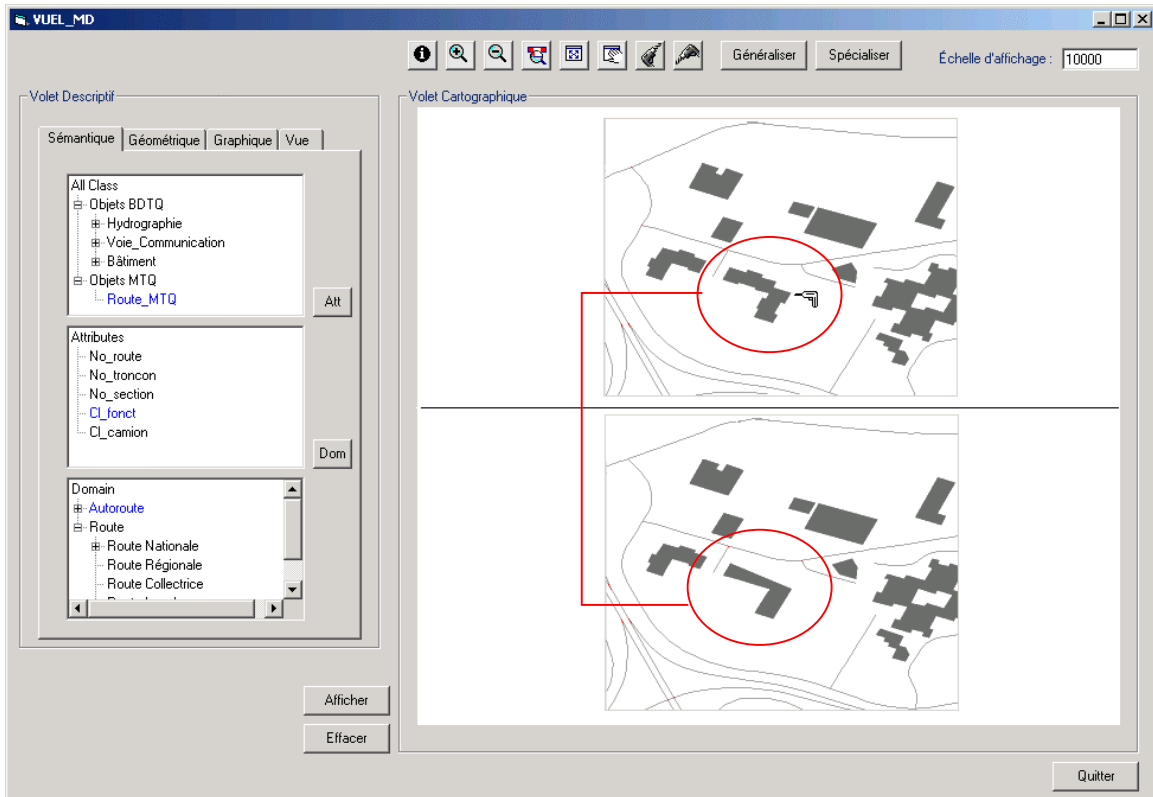


Figure 13 *The SOLAP interface*

The geometric tab allows a user to select the geometric level of granularity to be used to display the selected objects. The graphic tab is used to display objects with different visual variables depending on their object class or the value of a specific attribute (for objects of the same class). Finally, the View tab allows a user to select predefined views for which the scale, the semantic, the geometric and graphic representations have been defined. After having displayed the selected objects, different navigation functionalities are available to the user. Among them, there are some traditional spatial operators (zoom in, zoom out, pan, etc.) and some SOLAP specific functionalities to navigate between the possible multiplicities. For example, a user can select a specific object and drill-down on its geometry to see the same object with a detailed geometry (figure 13)

5. CONCLUSION

The concept of VUEL has been defined to add flexibility to the management of multiple views of a same dataset and to support geometric, semantic and graphic multiplicities. As such, we believe that it enhances the concept of multiple representations and offers a different, elegant theoretical framework. This basic construct has a formal definition, goes beyond the geometric elements of spatial databases and relies on multidimensional data structures. A first prototype using the vuel

concept has shown promising results and other challenges. In a near future, a web-based prototype with additional functions will allow us to further test the concepts presented in this paper, to optimize our data structure for better performance and to perform Map-on-Demand as well as SOLAP applications.

ACKNOWLEDGEMENT

The authors wish to acknowledge the financial support of the GEOIDE Network of Centers of Excellence via the DEC#9 project: *Development of automated techniques to extract, generalize, and access geospatial information from hyperspatial remotely sensed data*. The authors also acknowledge the financial support of the Natural Sciences and Engineering Research Council.

BIBLIOGRAPHY

- BÉDARD Y.**, 1997, Spatial OLAP, Vidéoconférence. 2ème Forum annuel sur la R-D, Géomatique VI: Un monde accessible, Montréal, 13-14 novembre
- BÉDARD Y.**, 1999, "Visual Modeling of Spatial Databases Towards Spatial Extensions and UML", *Geomatica*, Vol 53, No2, pp. 169-186
- Bernier E.**, 2002, "Utilisation de la représentation multiple comme support à la génération de vues de bases de données géospatiales dans un contexte SOLAP", Mémoire de M.Sc., Faculté de foresterie et de géomatique, Université Laval, 95p.
- CLARAMUNT, C. & M., MAINGUENAUD**, 1995, Spatial View: A dynamic and flexible vision of GIS database, Proceedings of the DEXA International Conference and Workshop on Database and Expert System Applications, Revell, N. and Min Tjoa, A. eds, Omnipress, London, UK, pp. 483-493
- HAN J., STEFANOVIC N. & KOPERSKI K.**, 1998, Selective materialization: An Efficient Method for Spatial Data Cube Construction, Proceedings of the 1998 Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'98), Melbourne, Australia, pp. 144-158
- MARTEL C.**, 1999, Développement d'un cadre théorique pour la gestion des représentations multiples dans les bases de données spatiales, Mémoire de M.Sc., Faculté de foresterie et de géomatique, Université Laval, 128p.
- THOMSEN E., SPOFFORD G., CHASE D.**, 1999, Microsoft OLAP Solutions, John Wiley & Sons, 509p.
- RIVEST, S., Y., BÉDARD, & P., MARCHAND**, 2001, Towards better support for spatial decision-making: Defining the characteristics of Spatial On-Line Analytical Processing (SOLAP), *Geomatica, the journal of the Canadian Institute of Geomatics*, Vol. 55, no. 4, pp. 539-555
- WEIBEL R., BERNIER E, BÉDARD Y. ET CECCONI A.**, 2002, "La généralisation à la volée ", dans : A. Ruas (edit), « Multiéchelle et généralisation », édition Hermes, to be published.