A FORMAL MODEL FOR THE SPECIFICATIONS OF GEOGRAPHIC DATABASES

Sébastien MUSTIÈRE, Nils GESBERT, David SHEEREN
COGIT Laboratory - Institut Géographique National
2-4 av. Pasteur - 94165 Saint-Mandé Cedex
France
{sebastien.mustiere, nils.gesbert, david.sheeren}@ign.fr

ABSTRACT

In order to capture and manage their geographic databases, database producers use relatively huge and complex documents: the specifications. Actually, these are the only sources of information describing precisely what is the content of a database, i.e. what is the meaning of each part of the data schema, which object are captured and how they are represented. As such, they are an important tool to describe, manage, distribute, transform and federate databases.

Unfortunately, even if they can be precisely organised, these specifications are usually expressed in natural language and thus in a poorly formalised language. Their manipulation is then a hard task, and their automatic manipulation is almost impossible. In this paper we claim that formalising existing specifications is an important task and we propose an object-oriented model in order to do that.

This model contains two main parts: a part formalising which objects of the real world should appear in the database, and a part describing how these objects are represented in the database. These model rely on a set of typical criteria encountered in database specifications: geometric criteria, relationship criteria, and nature criteria.

KEY WORDS

Database, specification, formalisation, model.

1 INTRODUCTION

As for any database, the content of geographic databases is described by the data schema and sometimes by metadata. Common geographic words are used to name the different part of the schema, like names of classes, attributes and relations in an object-oriented model.

But, for each database, there exist an exact meaning beyond these words. For example, if a class is named *River*, it may actually designate only permanent river in a database, or only natural river excluding human-made canals in another database. More, the name river may designate only rivers that are wider than 10 meters. Indeed, a database is associated to a certain level of detail, and only relevant objects are captured.

How a real world object is represented in the database is also a precise thing for each database. For example, the geometric line describing the river may be the axis of the river or one of its border.

All these definitions are stored in particular documents: the database specifications. These specifications are first used by database producers to develop precise guidelines for data capture. They can also be distributed to data users, in order to help them to understand in detail what the database contains.

Sometime, different specifications can exist for one database: the *content specifications* describe which objects of the real world should be represented in the database, and the *specifications of data acquisition* which detail the process for data capture into the system. Furthermore, subsets of specifications are usually provided to users along with the delivery of the spatial databases.

Actually, these documents are the only sources of information describing precisely what is the content of a database. Metadata may also exist, but in practice they are mainly used to describe global criteria like the reference system used, the geographic extent, the date of the capture, the source for capture, some evaluations of precision, and so on. They do not describe the exact semantic of each concept used in the database.

Unfortunately, even if they can be precisely organised, specifications are usually expressed in natural language and thus in a poorly formalised language. Their manipulation is then a hard task, and their automatic manipulation is almost impossible.

In the next section, we explain some possible use of more formalised specifications. Then, we propose a generic model of specifications for geographical databases. In section 3, we explain how to link data schemas to our model of formal specifications. In section 4, we detail basic elements of the model. This model has been defined from the study of existing specifications used in IGN, the French National Mapping Agency.

Before that, let us make a remark about the notion of ontology that receive more and more attention in the GIS community. This work may be related to the work on spatial ontologies. But, due to the numerous meanings of this term, we prefer to avoid it. If an ontology is thought of as a way to represent the intended meaning of a vocabulary [[1],[2]], this work is about the development of an ontology from database specifications. But if an ontology is thought of as a way to express the links between different vocabularies in order to facilitate the understanding between systems (in other words, a kind of thesaurus) [[3]], this work is not about the direct development of ontologies, even if it may take advantage of them. Indeed, our purpose is really a formalisation of specific definitions encountered in each database specifications.

2 ON THE UTILITY OF FORMALISING SPECIFICATIONS

In term of quality, it is recommended that the specifications present some characteristics. Among these, [[4]] mention that the specifications must respect all expressed needs, avoid useless details and must be clear, precise and complete, without ambiguities. But specifications are generally represented in natural language and described in a relatively informal way, dependent on the considered database. Moreover, specifications associated with different databases may be not organised in the same way. This lack of formalisation raises some important issues:

- it makes sometimes long to browse specifications in order to find a special piece of information;
- it leads to different interpretations due to the imprecise descriptions;
- it complicates their comparison due to the heterogeneity of their description;
- it induces some difficulties for automatic exploitation of specifications.

This issues have consequences on different tasks in the GIS area.

2.1 DATA ACCESS

The first task data users are confronted with is data access. Beyond technical and commercial issues, users must first be able to understand the content of a database in order to know if it is adapted to answer to their needs: this the notion of "fitness for use". They must be able to answer to questions such as "Does this database contains the information I am looking for?", "Are these objects described with an adapted level of detail, or is it over or under-detailed ?", "Between these two databases, which one is the more adapted to my needs?", "What should I filter in this database in order to make it simpler and more adapted to my needs", and so on.

All this tasks necessitates to understand clearly what is the meaning of the elements of the database. But present metadata can not fulfil these requirements as they usually describe the databases in a too global and limited manner.

In this context, the more the specifications are clearly organised, and the more they are homogenised, the easier it will be for users to understand and compare databases.

More, in the long run, we may be able to develop decision support system able to advise users with suitable data for their needs. Such a system can only be developed if, on the one hand, we are able to formalise user needs and, on the other hand, we are able to formalise what a database contains and for what it can be useful. Formalisation of specifications is thus one of the bricks necessary to build such a system.

2.2 DEVELOPING AND MANAGING DATABASES

From the data producer point of view, a more formal model of specifications could also help database designers to better define their specifications. As any model, this model is of course constraining and may be a limitation to the possible specifications that can be defined. But these limitations provide also a framework to help the definition of specifications more organised, better understood and less ambiguous.

A common model for all specifications produced by one data producer (which is actually not the case) has also the advantage of facilitating the exchange of knowledge in the company.

Another advantage of such a model is to facilitate the maintenance of specifications. For example, if specifications are updated, a model will allow to point directly at the part of the specifications that have changed when one must inform database users and producers.

More globally, we believe that working on a model of the specifications framework is good way to analyse deficiencies of current databases to better define them.

2.3 UNIFYING DATABASES

Many geographical databases exist to represent a same part of the world, seen at different levels of details and with different points of view. Unfortunately these databases are relatively independent, and there exists a growing need for the unification of different databases into a single one making explicit the relations between them [[5],[6],[7],[8]]. This unification can in particular lead to a database managing multiple representations [[9],[10]]. Such unification would help to:

- Maintain the databases and propagate updates [[11]]
- Perform some quality analysis, by using one database to control another one or by identifying inconsistencies [[12],[13]].
- Increase the potentiality of applications development from these databases. Some applications can take advantage of using databases with multiple representations [[14],[15]].

An important issue for the unification is the ability of understanding what are the differences between the databases, which first requires to understand what exactly contains each database. These informations are actually described in the specifications. Their formalisation should allow, as we explained for the issue of data access, to better understand separated databases. Additionally, it should facilitate their comparison, as the specifications will be defined with a common model.

A relatively automatic processing of these specifications appears also very useful. These specifications, once presented in a formal model, can be used to guide an integration process of several spatial databases. In such a process, their automatic exploitation could simplify the research of the correspondences between them. First, their description in a formal language could reduce their lack of precision and therefore the inconsistencies of representations between the databases. Second, their formalisation allows displaying them to the users in a more organised way. Then, their formalisation should allow an automatic comparison of them. For example, one could automatically determine that the specifications of the class *Stream* in one database are more restrictive than the specifications of the class *River* in another database. Thus, we know that each object of the class *Stream* should have a corresponding object in the class *River*, but the reverse is not true. More precisely, we can determine how the classes correspond. For example we could determine that *River* and *Stream* correspond to each other only when the length of the river or stream is greater than 200 m, or when the river or stream is permanent.

3 LINKING A FORMAL MODEL OF SPECIFICATIONS TO A DATA SCHEMA

In the preceding section we claimed for a formalisation of specifications. In this section and the next one we propose a model for this formalisation. This model has been developed from the study of actual specifications taken out of two databases from the IGN (French National Mapping Agency): BDTopo (a topographic database with a metric resolution) and BDCarto (a road network oriented database with a decametric resolution). This model is expressed in UML. Let us notice that, for testing the feasibility of the model, we instantiated in the XML language from the river and road network of the databases.

3.1 A METAMODEL

Our specification model is linked to the metamodel of the geographical database, represented in a simplistic but quite generic way by three classes: "Class," "Attribute" and "Association" (see figure 1). We do not, for the sake of compatibility, use more details about it since we need to be able to handle databases in any format.

Let us notice that this model is a metamodel where the classes of the database (road, river, etc.) are thought of as instances of the metaclass *Class*. Similarly the attributes and associations are thought of as instances of the metaclasses *Attribute* and *Association*. The specifications about these classes, attributes and associations are expressed through the other elements of the model, described hereafter.

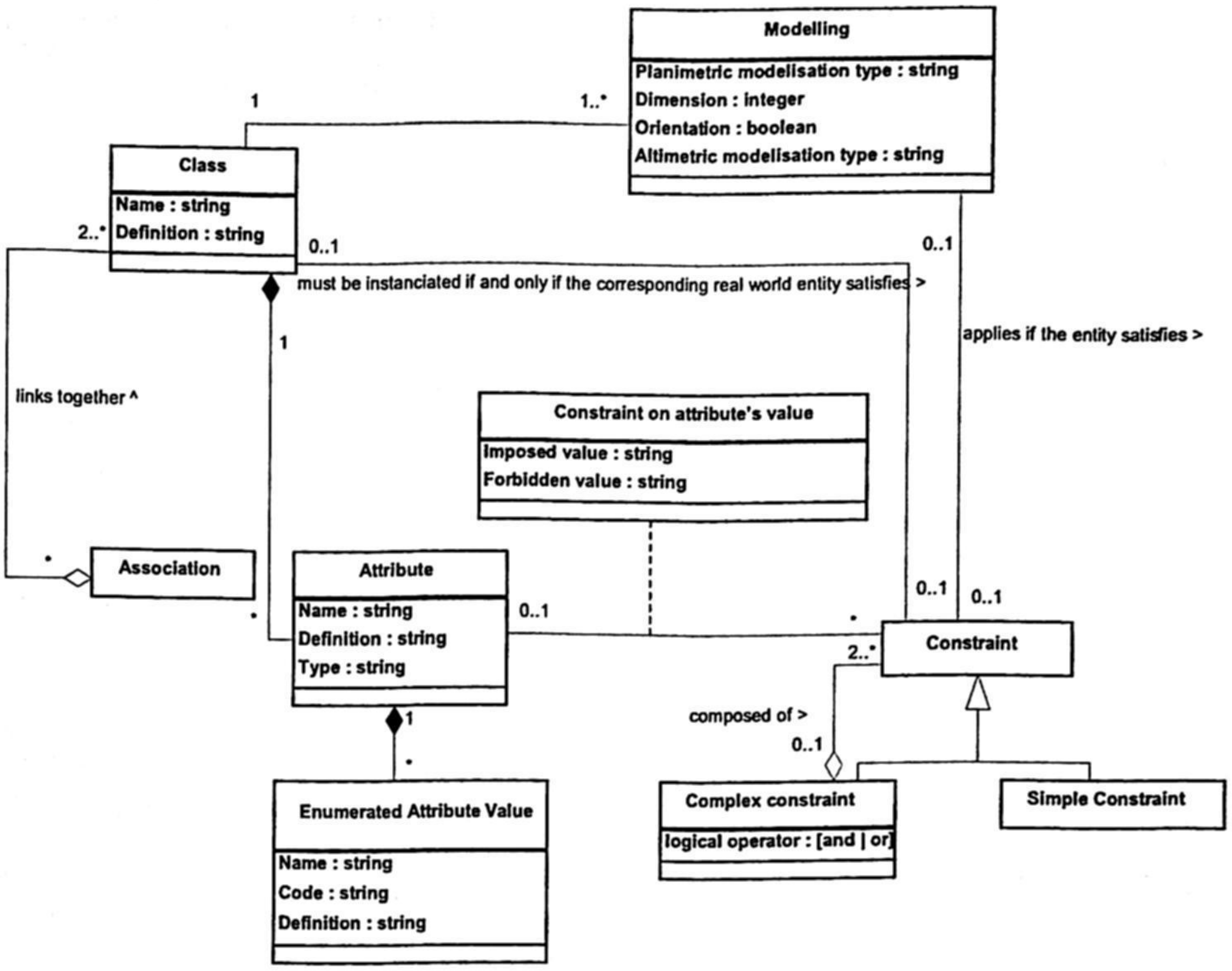

Figure 1. An extract of the general specifications model: the meta-model of the spatial database with the existence constraint, the modelling constraint and the constraint on attribute's value.

3.2 CONSTRAINTS OVER REAL WORLD OBJECTS

The main part of the model is a class hierarchy that allows to express a condition concerning a real world entity, such as "being greater than 100 meters" or "being inside a urban area". It consists of a base class "Constraint" and its subclasses.

It is important to notice that these "constraints" are not classical integrity constraints for the database. They are not constraints on the objects of the database (like "this attribute must be within such range of values, or an object of this class can not intersect an object of this class...). These constraints are used to express conditions over real world object, like "a *real world* river is represented in the database, if it is more than 10 meters wide".

To represent every possible condition that a real world entity could be asked to satisfy, we specialise the class "Constraint" into "Simple constraint" and a recursive "Complex constraint" which is composed of constraints linked together by a logical operator: *AND* or an inclusive *OR*. This way, we get a tree structure whose nodes are complex constraints, containing logical operators and linked to their children through the "is composed of" relation, and whose leaves are simple constraints. This allows to express complex constraints such as "being inside a urban area and either longer than 100 meters or wider than 30 meters," which is a complex constraint composed of a simple constraint and another complex constraint, in turn composed of two complex constraints. Simple constraints specialise in turn into three subclasses "Geometric constraints," "Relationship constraints" and "Nature constraints". Before describing these constraints more in detail in section 4, the following section describes how the constraints are used to express specifications on the geographic database.

3.3 WHAT AND HOW TO REPRESENT IN THE DATABASE: EXISTENCE AND MODELING

The class of constraints over real world object is the pivot class to express specifications. These constraints are linked to elements of the metamodel by several ways, either to express existence constraints, modelling constraints, or attribute constraints.

Constraints are first used to define the necessary and sufficient condition under which a real world entity should be represented in the base, like "*forests are taken into account in the database if and only if their area is greater than 8 ha*." This condition is called the "existence constraint" of the class, and is linked to it through the "must be instantiated if and only if..." association (figure 1).

Another purpose of the constraints is to indicate which modelling applies in which case when a class can accept several ones, as in "*a building is represented by its ground surface if it is more than 50 m² large, else its centre point is used*." This kind of constraint is named a "modelling constraint" and is linked to the "Modelling" class (figure 1). To indicate the way the real world entity's geometry is represented in the database, we use the "Modelling" class, whose attributes allow to specify, between others, the spatial dimension of the database object and the way its geometry is deduced from the real entity's one (for example "outline of the roof" or "ground surface"). A class may have several possible modellings, in which case each one should be linked to a modelling constraint in order to indicate how is chosen among them. For example, let the following specification for the "*River*" class be: "*a river is represented by its axis if it is less than 20 m wide, else it is entered as a surface*." We get two instances of "Modelling," one with *dimension* "line" and *planimetric modelling* "axis," linked to the geometrical constraint "*width less than 20 m*," and the other one with *dimension* "surface" and *planimetric modelling* "border," linked to the opposite constraint "*width more than 20 m*."

Constraints can also be used to express constraints on attribute values. Attributes are given a name, definition and type, which is generally sufficient. For example, "*number of lanes: integer*" does not *a priori* need to be further detailed. But there are cases however where it is not sufficient. Sometimes it is necessary to define how the attributes are instantiated. For examples constraints can be used to express specifications such as *"attribute 'Accessibility' of the class 'Road' is set to 'Impossible' if the width is less than 2 meters or if the road is unpaved"*

4 BASIC ELEMENTS OF THE FORMAL MODEL

We defined a constraint model, in order to formalise the content of the constraints, and facilitate their comparison (see figure 2). The elements of this part of the model are described in the following sections.

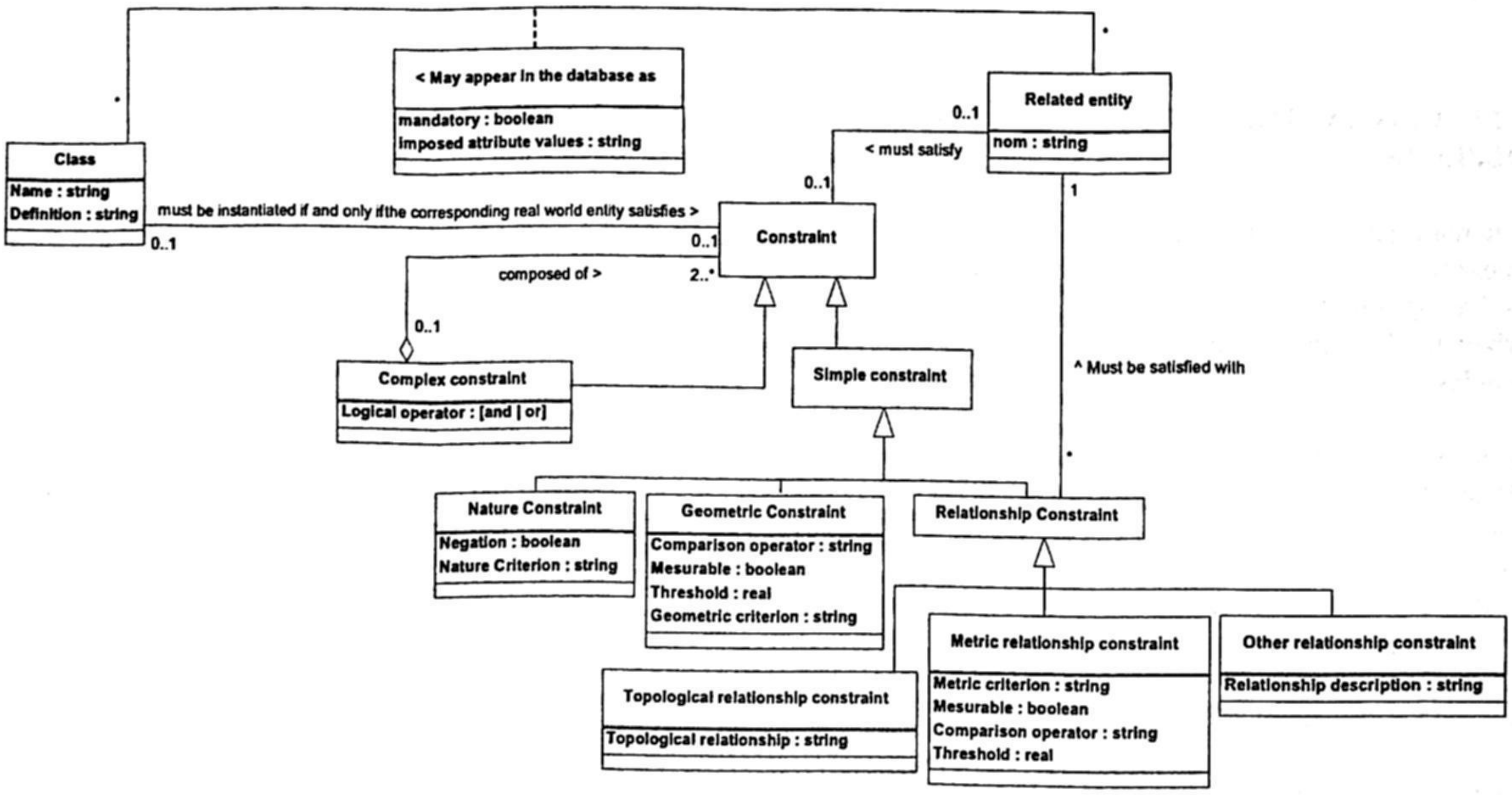
< May appear in the database as
mandatory : boolean
imposed attribute values : string
Class
Name : string
Definition : string
must be instantiated if and only if the corresponding real world entity satisfies >
0..1
0..1
Related entity
nom : string
0..1
< must satisfy
0..1
1
Constraint
composed of >
2..*
0..1
Complex constraint
Logical operator : [and | or]
Simple constraint
^ Must be satisfied with
Nature Constraint
Negation : boolean
Nature Criterion : string
Geometric Constraint
Comparison operator : string
Mesurable : boolean
Threshold : real
Geometric criterion : string
Relationship Constraint
Metric relationship constraint
Metric criterion : string
Mesurable : boolean
Comparison operator : string
Threshold : real
Other relationship constraint
Relationship description : string
Topological relationship constraint
Topological relationship : string

Figure 2. Extract of the elementary constraints of the geographical specifications model

4.1 GEOMETRIC CONSTRAINTS

"Geometric constraints" allow to specify a geometric criterion (length, area, etc.) and to demand that its value be above (or below) a given threshold. These constraints are commonly met, either as existence constraints or as modelling constraints.

These constraints are the easiest to formalise, as they are all relatively similar and clearly defined. The set of metric criteria used is restricted and a predefined set of possible values for the "geometric criterion" can thus be found to cover most cases: length, width, area, height. These constraints are also easy to compare, as no further knowledge is needed to know that "the width must be more than 20 meters" is more restrictive than "the width must be more than 10 meters".

4.2 NATURE CONSTRAINTS

"Nature constraints" are constraints specifying the nature of the entity. For example, we can describe there a condition like "*the building must be a house*" or "*the forest must be composed of conifers*".

It is difficult to formalise these constraints more than by just expressing the fact that the entity *is* or *is not* something. Thus, without additional knowledge, it is difficult to compare these constraints, except that some constraints are the opposite of some others. In order to go further, one would have to use some thesaurus of geographical terms [[3],[16]]. Those thesaurus can express that the words "unpaved road" and "track" are partly similar, or that "house" is a kind of "building," or that "lac" is the French translation of "lake." These knowledge can be of great help to compare nature constraints, even if some ambiguities can appear, as certain identical terms can be understood differently in different databases.

4.3 RELATIONSHIP CONSTRAINTS

"Relationship constraints" allow to handle conditions that do not bear on the entity itself but rather on the surrounding ones. For example we could have a constraint like: "*a path is taken into account only if it leads to a house.*" Among these relationship constraints, we further distinguish metrical relationships, like "*to be more than 50 m away from a house,*" and topological relationships, like "*to be inside a forest,*" which are common. To specify with what the relationship is supposed to occur, we use the dedicated class "Related entity" that designated real world entities (and not database entities). If giving the name of the related entity does not suffice, for example if we do not want it to be any kind of road but to be a surfaced one, we link it once again to another instance of "Constraint" expressing "to be a surfaced road."

One frequently encountered special case is when the specifications' text directly refers to the database rather than to the real world, as in: "*culs-de-sac are taken into account if they lead to an object to the class of of theme A (particular buildings).*" In this case, we still create an instance of "Related entity" (named *particular building*) and we link it to the mentioned database classes (those of theme A) by the "May appear in the database as" association class with "mandatory" attribute set to true. In other cases, in order to facilitate future automatic processing of the data, we also link the entity to the database classes that may represent it (if there are some), but we then set the "mandatory" attribute to "false" to express that this link is only informative and not really part of the specifications. In some other cases, the related entity may not be linked at all to the database classes, if no class is used to represent this entity, as in the constraint "*being outside the town*" if the concept of town is not represented in the database.

These kind of constraints may be very precisely formalised if we use some models of spatial relations to describe them, like models to describe topological relations [[17],[18]] or distance and orientation relations [[19]]. These constraints may also be compared by using some models of conceptual neighbourhood between relations [[20],[21]].

4.4 OTHER CONSTRAINTS TO BE INTEGRATED IN THE MODEL

This classification of constraints was not established of course immediately, due to the above-mentioned lack of formalism in the specifications. In order to validate the model, we then instantiated it on some excerpts of our two sets of specifications. Instantiation of the model brought to light two constraint types which had not been taken into account: *representativeness constraints*, and *constraints relative to attribute value changes.*

Representativeness constraints play almost the same role as existence constraints, except that they apply when the database object does not correspond to a single real world entity but is rather meant to be representative of a group. At the moment, we distinguish two subcases: there is one single object which represents the group (aggregation), or there are several of them (representative elements), in which case there is generally no one-to-one or even one-to-several possible mapping between database objects and real world entities, only a group-to-group relation. An example of aggregation is this excerpt of specification regarding the class "*Reservoir*": "*if there are many reservoirs close together whose diameter is less than 10 m, then they are merged.*" An example of representative elements could be buildings in a city in a generalised database.

Constraints regarding attribute value changes generally appear as geometrical ones: "*attribute changes for the* *road section class are not allowed for a length less than 100 meters.*" These constraints are neither existence constraints explaining which objects should be represented, nor attribute constraints explaining how to set the value of the attributes. They describe when to take into account the changes of attribute to determine the number of database objects to be created along the road.

These new constraints should now be integrated into the model, whose validation is going on. We did not either study in depth specifications and constraints concerning database associations.

5 CONCLUSION

In this paper we presented a model for formalising specifications of geographic databases. This model has two main parts: a part to define which object of the real world are represented in the database, and a part to define "how they are represented. This model rely on a set of predefined constraints on real world objects, like geometric, relationship and nature constraint.

This model is still under development and is part of an ongoing work. We intend to use it in a process of unification of databases. First, the model is used in the context of matching data schemas. Indeed, it is necessary to compare specifications in order to match precisely schemas. Second, the model is used in the context of analysing differences between databases [[22]]. Indeed, it is important to differentiate between "normal" differences justified by the differences in the specifications, and "wrong" differences due to errors in the databases.

More globally, we believe that database specifications are rich documents insufficiently used. We hope that going in the direction of their formalisation is a good avenue to increase their utilisation.

ACKNOWLEDGEMENT

We thank Anne Ruas who suggested the exciting idea of working on the formalisation of geographic databases specifications.

REFERENCES

[1] Guarino N. 1998. *Formal Ontology and Information Systems*. IOS Press, Amsterdam.

[2] Fonsesca F.T. and Egnehofer M.J. 1999. Ontology-driven geographic information systems. In *Proceedings of ACM GIS'99.*

[3] Smith B. and Mark D.M. 2001. Geographical categories: an ontological investigation. In *International Journal of Geographical Information Science, vol.15,* n.7, pp.591-612.

[4] Fougères A.-J. and Trigano P. 1999. Construction de spécifications formelles à partir des spécifications rédigées en langage naturel, In *Document numérique*, 3(3-4), pp. 215-239

[5] Devogèle T., Parent C. & Spaccapietra S. 1998. On spatial database integration. In *Int. Journal oj Geographical Information Science*, vol.12, n.4, pp.335-352.

[6] Weibel R. & Dutton G., 1999. Generalising spatial data and dealing with multiple representations. In *Geographical Information Systems – Principles, Techniques, Application and Management, 2nd Edition*, Longley P.A., Goodchild M.F., Maguire D.J. & Rhind D.W. (eds), Vol.1, pp.125-155

[7] Vangenot C., 2001. *Multi-représentation dans les bases de données géographiques*. PhD. Thesis, Ecole Polytechnique Fédérale de Lausanne.

[8] Ruas A. 2002. Pourquoi associer les représentations des données géographiques?. In *Généralisation et Représentation Multiple*, Ruas (ed), Hermes-Lavoisier, pp.45-54.

[9] Bertolotto M., De Floriani L., Marzano P., 1995. A Unyfying Framework for Multilevel Description of Spatial Data, Spatial Information Theory – A Theoretical Basis for GIS. In *Lecture Notes in Computer Science, n.988*, A.U. Frank, W. Kuhn (Editors), Springer-Verlag, 1995.

[10] Rigaux P. and Scholl M. 1995. Multi-scale partitions: Applications to spatial and statistical databases. 4th International Symposium on Advances in Spatial Data, SSD'95, pp.170-183.

[11] Lemarié C. and Badard T. 2001. Cartographic database updating. In *Proc of* International Cartographic Conference (ICA 2001), vol. 2, pp.1376-1385.

[12] Egenhofer M.J., Clementini E. & Di Felice P. 1994. Evaluating inconsistencies among multiple representations, In *Proceedings of the Sixth International Symposium on Spatial Data Handling* (SDH'94), Edinburgh, Scotland, pp. 901-920.

[13] Bel Hadj Ali A., 2001. Positional and Shape Quality of Areal Entities in Geographic Databases - Quality information aggregation versus measures classification. In *Proc of ECSQARU Workshop on Spatio-Temporal Reasoning and Geographic Information Systems*, Toulouse 2001.

[14] Car A. and Frank A. 1994. Modelling a Hierarchy of Space Applied to Large Road Networks. In *Proc. oj Int. Workshop on Advanced Research in Geographic Information Systems*, Springer-Verlag, pp.15-24

[15] Yuan M. and Albrecht J. 1995. Structural Analysis of Geographic Information and GIS Operation from a User's Perspective. In *Proc. of COSIT: Spatial Information Theory*, pp.107-122.

[16] Kavouras M. and Kokla M. 2002. A method for the formalization and integration of geographical categorizations. In *Int. Journal of Geographical Information Science*, vol.16,no.5,pp.439-453.

[17]. A Topological Data Model for Spatial Databases. In *Proc. of 1st Int. Symp. on the Design and Implementation of Large Spatial Databases*, LNCS 409, pp.271-286. Springer-Verlag.

[18] Worboys M.F., 1996. Metrics and topologies for geographic space. In *Advances in GIS research II : proc. of 7th International Symposium on Spatial Data Handling*, Kraak and Molenaar (eds), Taylor and Francis, p.365-375.

[19] Hong J.-H. 1994. *Qualitative Distance and Direction Reasoning in Geographic Space*. PhD. Dissertation, University of Maine.

[20] Bruns H.T. and Egenhofer M.J. 1996. Similarity of Spatial Scenes. In *Proc. of 7th Int. Symposium on Spatial Data Handling* (SDH), Taylor and Francis, pp.173-184.

[21] Paiva J.A. 1998. *Topological equivalence and similarity in multi-representation geographic databases*. PhD Thesis in Spatial Information and Engineering, University of Maine.

[22] Sheeren D. 2002. L'appariement pour la constitution de bases de données géographiques multirésolutions - vers une interprétation des différences de représentation. In *Revue Internationale de Géomatique*, Vol.12, n.2/2002, pp.151-168.