

Formalisation of Geographical Database Specifications

Nils Gesbert

Institut Géographique National, 2/4 avenue Pasteur, 94165 Saint-Mandé Cedex,
France
`nils.gesbert@ign.fr`

Abstract. The specifications of a topographical database, like those national mapping agencies produce, contain a lot of information about the data, which is needed to correctly and precisely interpret their meaning. This information, which is not present in the raw data model, is in particular crucial to data interoperability and geographical database integration.

But the specifications are huge text-based documents which are ill-suited to automatic processing. In particular, it is quite difficult for the time being to do an accurate comparison of two specification sets. This article addresses this problem by defining a method to represent these specifications in a formal and unified way. The main idea of it consists in defining a model of real ground entities and linking it to the actual data model of the database. Specifications are seen, in object-oriented terms, as relationships between a class of real ground entities and one or several classes of database objects.

1 Introduction

A particularity of topographical databases, as opposed to most kinds of classical databases, is the fact that the exact part of the real world they are supposed to represent is defined only quite fuzzily. Indeed, usually, the conceptual schema of a database is enough to exactly know what is to be captured in the database and what is not. For example, if some factory has a database of the articles it produces, one knows that this database contains a record for every article — or perhaps only for every article of a given, well known type; and that if it does not contain some article, it is probably an error.

This is not so in topographical databases. Even if the conceptual schema of the database says that it contains buildings, it does not mean that it contains every building. In fact, it contains only buildings that are *important enough*, or that are meaningful for the described landscape. And this is the same for every kind of entity represented in the database, because a landscape is something too complex and too quickly evolving to be exhaustively represented in a database: you have to filter it in some way.

The fuzziness here lies in “important enough.” How do you know if something is important enough to be represented in the database? It is to answer this question that the databases come with content specifications in addition to their —

usually object-oriented — conceptual schema. These are huge, text-based documents that detail the so-called “selection criteria” for each class of the schema, which restrict the set of entities involved by the class. In addition to this, they also describe the meaning of the classe’s attributes, and the way the geometry of the entity is represented in the database.

The information contained in the specifications is thus necessary for uses of the database that rely on a very precise interpretation of the data, especially as soon as you need to know what sort of entities are *not* present in the database; like if you want to know, on a map, to what the white parts may correspond on the real ground. One such use is integration of several topographical databases: it is necessary for it to thoroughly compare the sets of specifications from each database.

Unfortunately, the text-based shape of the specifications makes them quite ill-suited for automatic or semi-automatic processing. Thus, it would be very useful to represent the information they contain, or at least a reasonable subset of this information, in a more formal way. This article aims at defining a formal model for representing the specifications of topographical databases, with the main purpose of facilitating database integration. In a first part, we will present the topographical databases, their particularities and their specifications in more detail. In a second part, we will present the main ideas for the specifications model. Finally, in the third part, we will describe the actual model and show some examples.

2 Topographical Databases and Their Specifications

2.1 Topographical Databases

We will use throughout this article the example of topographical databases produced by the French National Mapping Agency (Institut Géographique National, IGN). These are vector databases, with an object-oriented conceptual schema. The main ones are a large scale database, BDTopo; a small scale one, BDCarto; and a road-oriented mixed scale one, Géoroute.

We mentioned in the introduction that topographical databases do not represent everything in their domain. Other particularities of this sort of databases come from this domain itself. [1] mention several remarkable properties of geographical concepts as opposed to other concepts: first, the location of a geographical entity is intrinsically tied with the entity itself, and it is impossible to separate “what” and “where.” Geometrical and topological properties, like the boundary of entities, are fundamental. Second, some entities exist only as products from human cognition, like bays or peninsulas; the authors describe them as “shadows cast by human reasoning and language onto the spatial plane.”

We can add to these the importance of scale or resolution. Indeed, depending of the resolution you observe the landscape with, some geographical concepts may be irrelevant. For example, you cannot tell the boundary of a forest with 1 m precision, and conversely you cannot see each tree individually at a resolution

of 20 m. Some other concepts are relevant at every considered resolution but their instances are different, for example a river might divide itself in several parts, forming islands, when you look at it with 100 m precision, but at 1 km resolution you would not see the islands, just one single river. In fact, it corresponds to what you would see in this landscape from different distances: like if you are just in front of a wall, what you see is bricks; but from a hilltop, you may see a town.

So we see that a topographical database is not directly a representation of the physical ground: rather, it is a representation of the “conceptualised ground,” which we can define in an abstract way as the set of geographical entities present on the considered landscape. The specifications of the database contain rules to extract from this conceptualised ground the set of *relevant* entities and to represent them according to the database’s nominal resolution. It is worth noting that, if topographical databases frequently represent every sort of entities with the same nominal resolution, this is mostly due to historical reasons: maps have only one single scale. But this does not need to be so and multi-resolution databases will probably be more and more frequent as time passes.

2.2 Database Specifications

As said before, topographical database specifications usually present themselves as large text documents. The conceptual schemas of the databases are generally object-oriented and there is a specification for each class. At IGN, they are always structured more or less in the same manner: for each class, there is a first part *definition*, which indicates what concept the class is meant to represent. Then a second part, *selection*, usually the most important in size, restricts the set of relevant entities. The third part, *geometric modelisation*, indicates how the geometry attribute of the object is generated from the entity it represents; the resolution always intervenes in this, either implicitly as the nominal resolution of the database, which is mentioned as metadata in the general specifications, or explicitly if the particular class does not use the general specification. Finally, an *attributes* part describes the thematical (non-geometric) attributes of the class.

Note that what the specifications deal with is not the actual content of the database, but rather its theoretical content, that is, what it would contain if there were not any errors. This “perfect database” is traditionnally called *nominal ground* at IGN. The specifications can be seen as describing the relation between the conceptualised ground and the nominal ground of the database (see figure 1). In our work, we intend to describe this relation in a formal way.

3 A First Approach of Specification Formalisation

To make things clearer, we will use the precise vocabulary defined by [2]: a *concept* is described by its *intension*, which possesses properties and an abstract classification mechanism allowing to tell which objects are and are not instances of the concept. For example, the “river” concept has properties such as navigability or width, and the classification mechanism comes from human experience:

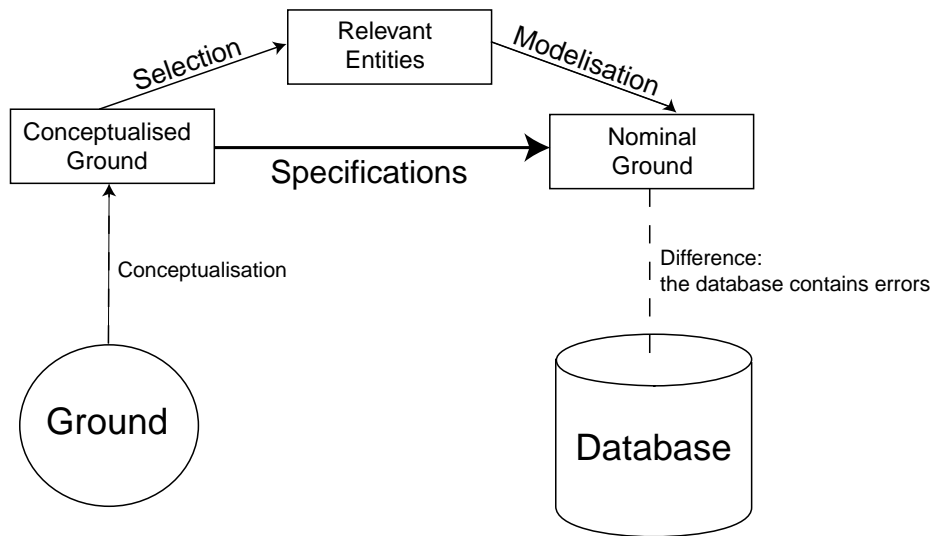


Fig. 1. The specifications do not directly describe the passage from the ground to the database but rather the passage from the conceptualised ground to the database's nominal ground.

we are able to recognize a river. A *class* is the extension of a concept, that is, the collection of all objects that pass the classification mechanism; so the “river” class would be the collection of all rivers. The term *occurrence* is used to specify an individual entity that might be placed into a class, while an *instance* refers to an occurrence that has been placed into a class. The authors furthermore distinguish *instanciation*, the creation of a *new* occurrence and its addition to a class (making it an instance), which corresponds to object creation in a database, from *classification*, the placement of an *existing* occurrence into a class, which corresponds to recognizing a real world entity as belonging to a given concept. In addition to this, we use preferentially the term “entity” when referring to the ground and “object” when referring to the database.

3.1 Formalising Selection Criteria

A first approach to formalising this specifications was taking the classes one by one, assuming the textual general definition described a concept whose instances would be clearly recognizable. We would then formalise the selection criteria for each class independently. The first step was the classification of these criteria into several kinds of constraints. We determined that three basic sorts of constraints, which may be freely combined with logical operators, are encountered in the specifications ([3]):

- geometrical constraints, generally taking the shape of a selection threshold on a geometrical property like length or width (entities below the given size are not captured into the database);
- nature constraints, bearing on properties of the considered entity other than its geometry;
- relationship constraints, bearing on the context of the entity (the entities near it, or the general type of landscape it is in), which we further subdivide into metrical relationships (being nearer from another entity than a given threshold), topological ones (being in, or overlapping, another entity), and other ones (for somewhat fuzzy notions like “lead to”).

3.2 Limits of the Class-By-Class Approach

But it appeared eventually that this approach was too limiting. There were three main problems:

- first, the classes from the conceptual schemas of the databases often group together several different concepts, and two different databases do not necessarily use the same groupings. For example, in BDCarto, the “river section” class contains aqueducts; and in BDTopo, aqueducts are represented but in the “canalisation” class, together with oil pipelines. Therefore, the class-based approach would have led to problems when applying our model to database integration.
- Second, it is somewhat convenient that the relationship constraints described above always refer to concepts that are themselves described by the model. But these concepts are often not represented by schema classes; this is the case in particular when they belong to a smaller scale than that of the database. For example, some selection criteria in the BDTopo (large scale) are not the same depending whether we are in a town or in the countryside, but towns are not represented in this database.
- Last, some classes from the database schemas cannot be consistently associated with a concept from the conceptualised ground, because their general definitions mix together real selection criteria bearing on the entities and representation constraints bearing on the database object itself. This is the case, for example, of the “river section” class. Indeed, the definition of this class, in both BDCarto and BDTopo, says that a river section is “homogeneous for its attribute values.” But such a definition do not allow one, in front of a landscape, to tell what exactly are the river sections, so there does not exist any consistent concept of river section. Instead, one has to use the concept of “river,” and tell that the river gets divided into sections at the modelisation step.

These three reasons lead us to create, in addition to the database schema, an independant schema for the concepts used in the specifications. An empirical method to find the relevant concepts is to browse the specifications for keywords. Then one organises these concepts into an object-oriented schema, and the specifications may be represented as relationships between the classes of this new schema and those of the database’s schema.

4 The Specifications Model

Thus, we will propose a model in three parts. The first part will be the present database schema; the second one will be the schema of concepts (describing the conceptualised ground), and the third will represent the specifications. We can note that the second part, as specification of a conceptualisation, is an “ontology” according to the definition from [4] (see [5] for the other possible meanings of “ontology”).

This structure is represented figure 2 as a UML profile (extension of the UML meta-model). There are three meta-elements, corresponding to the three parts of the model. As a meta-model, it allows to generate, through instantiation, these three parts. Let us describe each of them.

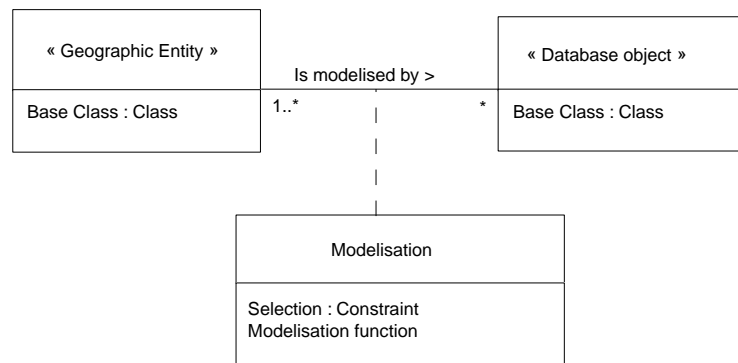


Fig. 2. UML profile describing the specifications model.

4.1 Database Schema

Instantiation of the “database object” metaclass just gives the classes from the database schema.

4.2 Conceptualised Ground Schema

The instances of the “geographical entity” metaclass represent the geographical concepts used by the specifications. They should be generic as far as possible, as the goal is to have only one conceptual ground schema usable for several databases (possibly by extending it).

These classes possess properties, which will be used for the modelisation specifications. These properties are not readily accessible as a universal value, as said above; rather, they always depend on the resolution. Here resolution can

intervene as a precision, for numerical properties (like “river width with 3 meter precision”), but may also indicate a generalisation level for more complex properties; for example, representation of curves on a road needs a special treatment for small scales. The “exact” width of the river or the “exact” shape of the road are abstractions which do not really exist. For this reason, in object-oriented terms, the properties of these classes are operations rather than attributes, resolution being a parameter of the operation. Also, the values of properties are allowed to depend on the considered position inside the entity, like width or navigability of a river may vary along its course. Such variable properties were proposed in the MADS model ([6]).

4.3 Modelisation Relationships

Instanciation of the association itself shows by which database classes a geographical concept may be represented; each concept may be represented by none to several database classes, and each database class participates in the representation of at least one concept. Then, the instance of the “modelisation” element contains the actual specification corresponding to the association: selection and modelisation. For the selection criteria, we use the constraints defined in section 3 (figure 3). For modelisation, the exact formalism is still to be defined, but it will assume that there exists a number of primitive operations that do not need to be specified and can be used to specify more complex ones. One such primitive would typically be “get the entity’s boundary at resolution [parameter].”

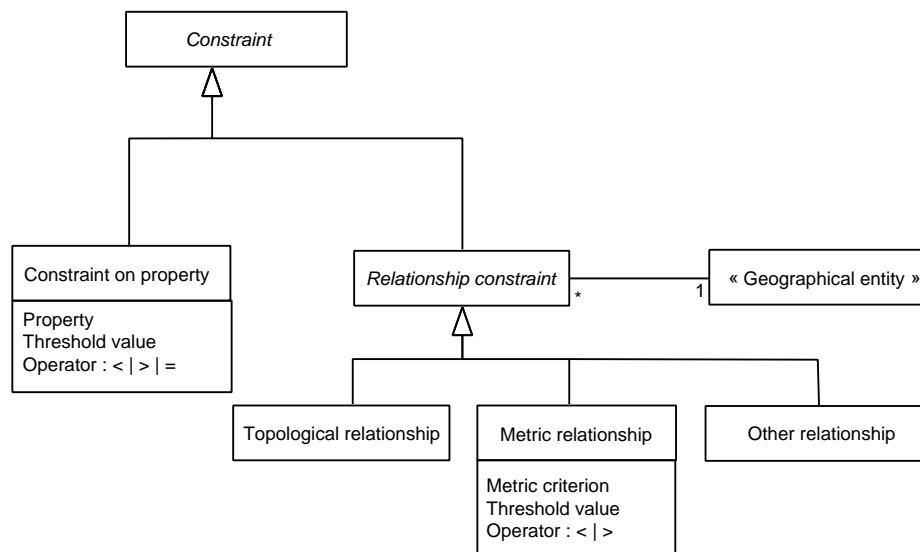


Fig. 3. The different encountered constraint types.

4.4 Examples

Two instantiations of our metamodel are shown on figures 4 and 5. The example is the hydrographic network; the ontology schema was first done for BDTopo and then extended to include the aqueduct, which in BDTopo does not belong to the hydrographic network. The dashed lines indicate the instances of the modelisation association. Selection criteria are not shown on the figures, but an example of it is “width > 7,5 m” on the association between the river (“rivière”) and the water surface (“surface d’eau”) in BDTopo. As there is no selection criterion on the association between the river and the river section (“tronçon cours d’eau”), we can see that when a river is wider than 7,5 meter it is represented by objects of both classes. Another example of selection criterion is “has a toponym” on the association between river and named river (“cours d’eau nommé”).

These schemas show clearly many design differences between the two databases, for example we can see that BDTopo, unlike BDCarto, does not have a network node (“nœud du réseau”) and that dams and waterfalls are represented as river sections. But it also shows that, beyond these differences, they both represent essentially the same entities. Note that concepts like confluence or river mouth (“embouchure”) seem not to be represented in BDTopo, but in fact they intervene in the modelisation specification for the river section (which is not shown), as places where to cut.

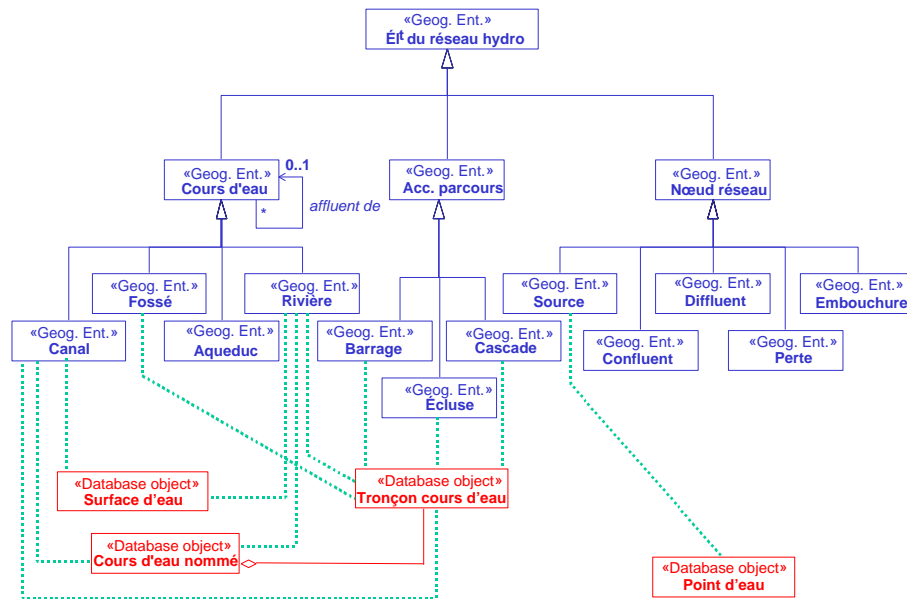


Fig. 4. Instantiation of our metamodel for the hydrographic network in BDTopo.

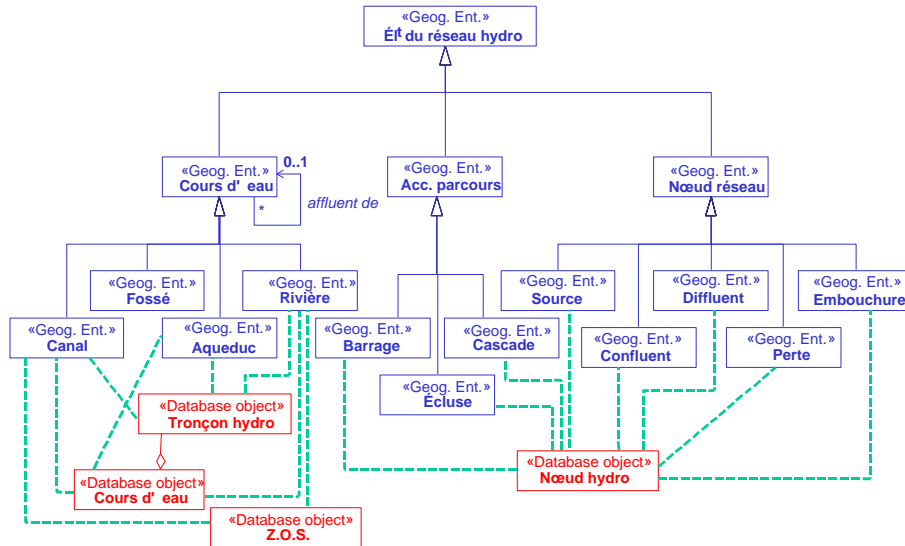


Fig. 5. Instanciation of our metamodel for the hydrographic network in BDCarto.

5 Conclusion

We proposed in this article a model for representing geographical database specifications, and their integration with the conceptual schema of the corresponding database.

We hope that such a model will allow a better understanding of the data, and in particular that it will make the comparison of several specification sets easier. The main application it is intended for is integration of several databases with different scales ([7]), which could lead to a multi-representation database ([8]). This problem has already been started on ([9]), but directly at the data level, that is, in terms of geometrical data matching. For a complete database integration, it is also needed to integrate the databases' conceptual schemas and do semantic matching. Ontology plays an important role in this step ([10]). Our model mainly aims at facilitating this semantic matching. It will also be used to find inconsistencies between different databases and errors in geometrical matching ([11]). Indeed, only the specifications can allow one to know if a difference in representation is all right and a consequence from the difference of point of view or if it is an error or a difference in data actuality.

References

1. Smith, B., Mark, D.M.: Ontology and geographic kinds. In Poiker, Chrisman, eds.: Proceedings of the Eighth International Symposium on Spatial Data Handling, International Geographical Union, Geographic Information Science Study Group (1998) 308–320
2. Brodaric, B., Gahegan, M.: Distinguishing instances and evidences of geographical concepts for geospatial database design. In Egenhofer, M.J., Mark, D.M., eds.: GIScience 2002. Number 2478 in Lecture Notes in Computer Science, Springer-Verlag (2002) 22–37
3. Mustière, S., Gesbert, N., Sheeren, D.: A formal model for the specifications of geographical databases. In Levachkine, S., Serra, J., Egenhofer, M., eds.: Semantic Processing of Spatial Data, proceedings of workshop GeoPro 2003. (2003) 152–159
4. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. In Guarino, N., Poli, R., eds.: Formal Ontology in Conceptual Analysis and Knowledge Representation. Kluwer Academic Publishers (1993)
5. Guarino, N., Giaretta, P.: Ontologies and knowledge bases : Towards a terminological clarification. In Mars, N.J., ed.: Towards Very Large Knowledge Bases. IOS Press, Amsterdam (1995)
6. Parent, C., Spaccapietra, S., Zimanyi, E., Donini, P., Plazanet, C., Vangenot, C., Rognon, N., Pouliot, J., Crausaz, P.A.: Mads : un modèle conceptuel our des applications spatio-temporelles. *Revue internationale de géomatique* **7** (1997)
7. Devogele, T., Parent, C., Spaccapietra, S.: On spatial database integration. *International Journal of Geographical Information Science* **12** (1998) 335–352
8. Vangenot, C.: Multi-représentation dans les bases de données géographiques. Thèse de doctorat, École polytechnique fédérale de Lausanne (2001)
9. Devogele, T.: Processus d'intégration et d'appariement de bases de données géographiques; application à une base de données routières multi-échelles. Thèse de doctorat, Université de Versailles (1997)
10. Partridge, C.: The role of ontology in integrating semantically heterogeneous databases. Technical Report 05/02, LADSEB-CNR, Padova (2002)
11. Sheeren, D.: L'appariement pour la constitution de bases de données géographiques multi-résolutions : vers une interprétation des différences de représentations. *Revue internationale de géomatique* **12** (2002) 151–168