Geomodeling and Implementation of a Geodatabase Using CASE-tools and ArcCatalog

Ward Verlinden and Eric Bayers
Department of Cartography, National Geographical Institute – Belgium

Abstract

“Seamless Geographic Information System of Reference” or SGISR is a development project of Belgium’s National Geographical Institute. One of the major goals of this project is the design and implementation of a centralized, spatially continuous GIS to store, manage, edit and distribute topo-geographic reference data. Based on a ISO19110 compliant Feature Catalogue, a Conceptual Data Model was developed to describe all object classes, attributes and relationships. This CDM was created with a CASE-tool, using UML class diagrams. Next, the CDM was translated to Esri’s geodatabase model, coupling all classes in the SGISR model to one of the basic ESRI classes. Tagged values were used to store information, specific to the geodatabase model. Finally, the resulting Logical Data Model was used to create a geodatabase. The model was exported to a XMI-file and validated using Esri’s “Semantic checker”. The “Schema wizard” was used to create the geodatabase.

Introduction

As Belgium’s NMA, the National Geographical Institute (NGI) is responsible for the production of geographical data for the whole of the Belgian territory. Based on this data, several topographical map series with scales ranging between 1:10,000 and 1:250,000 are produced.

In 2001, NGI Belgium completed the first numerically produced edition of the 1:50,000 map series. The first numerically produced edition of the 1:10,000 map series was planned to be completed in another couple of years time. Until then the focus had been on the production of a separate vector data set for each cartographic product. Now however, NGI Belgium was faced with the challenge of systematically updating its vector data sets.

At the same time, it was clear that the production of topographic maps was no longer the only important use of these data sets. The use of NGI vector data in GIS applications was steadily growing. This evolution called for a new way of structuring the data, which focused less on cartography and more on offering a geographically oriented model of the real world.

Another important decision was to move away from the principle of two parallel production lines, resulting in two separate data sets. It was felt that there should be one centralized process to update a single geographical database containing reference data. This reference data would then be used for several applications – cartographic and other – at different scales. This way, updating information would be collected once and used many times.

To develop these ideas into a solution for the needs of NGI Belgium, a project called Seamless Geographic Information System of Reference (SGISR) was started. Three main objectives were determined for SGISR:

1. Design of one integrated production line to collect, manage and update topo-geographic reference data.
2. Implementation of a centralized and spatially continuous GIS to store, manage, edit and distribute topo-geographic reference data.
3. Development of the necessary tools for the realization at NGI Belgium of applications with a conceptual scale between 1:10,000 and 1:50,000, based on topo-geographic reference data.

At the core of SGISR is the development of a spatially continuous geographical database to store topo-geographical reference data. This paper discusses the different phases of the data model design.

Conceptual modeling

In a first phase of the conceptual modeling process the existing data sets, used for the map series 1:10,000 and 1:50,000, were analyzed as a starting point for determining the geographic content, necessary for the SGISR database. Because both data sets were originally designed for very specific cartographic applications, it was clear that modifications in terms both of content and data structure would be necessary.

For some types of information, especially for networks which are partially underground (e.g. high tension lines), the existing data sets were not complete because the underground parts don't feature on the topographic maps. In most of these cases it was decided to incorporate the whole of the network, including any underground parts, into the new data model. A number of attributes were also added to existing elements to describe additional characteristics. Domains for existing attributes were extended to provide more detailed information. In a limited number of cases some completely new features were added to the geographical content that was already present.

Changes in data structure were mostly linked to the shift from a cartographical to a geographical point of view. The original structure of the existing data sets had been heavily influenced by the cartographical production process. While a restructuring effort a few years prior to the start of SGISR had already solved many of these problems, some issues still needed to be addressed to eliminate all purely cartographic aspects from the data model.

Based on all these content and structure related considerations a feature catalogue was compiled, defining all feature types, their attributes and relationships for the new data model. This feature catalogue was designed to comply with the international standard for feature cataloguing, ISO 19110, which at that time, was only available as a draft version (Anon. 2001).

All mandatory and some optional elements of the ISO 19110 template were used. In addition, certain elements were added to adapt it to the specific context of NGI Belgium and the SGISR project. For purely practical reasons the feature catalogue was split into eight domains, each domain containing a part of the model that describes a specific theme, for instance the road network. A field was also added to enter a geometrical primitive for each feature type into the catalogue. The most important adaptation to the template was made to create a bilingual feature catalogue. Because of the importance of both Dutch en French languages in Belgium, it was decided that all names and definitions of feature types, attributes and relationship types should be available in both of these languages.

The SGISR feature catalogue is stored in a small Access database. A data input form and a customized reporting tool were added to this database, both of which make use of the data model domains to enable the user to focus on one specific subset of the model at a time.

Alongside the feature catalogue some preliminary conceptual class diagrams were drawn up, using UML (Unified Modeling Language). These diagrams helped the project team understand and communicate the significance of each feature type and were instrumental in visualizing the different relationship classes that linked the feature types to one another. Based on these preliminary diagrams a complete Conceptual Data Model (CDM) was created, integrating all the information of the feature catalogue. A CASE tool (Computer Aided Software Engineering) called Visio was used to draw the CDM. A UML object class was created for every feature type, containing the feature type's name, the names of the attributes and the attribute data types. For attributes with specific attribute value domains, data types defining these domains were added to the model. Relationships were modeled using binary association, composition and generalization relations. While most relationships between feature types were modeled using binary associations, characterized by their name and cardinalities as defined in the feature catalogue, in a limited number of cases composition relations were used to indicate strong part-whole relationships between feature types.

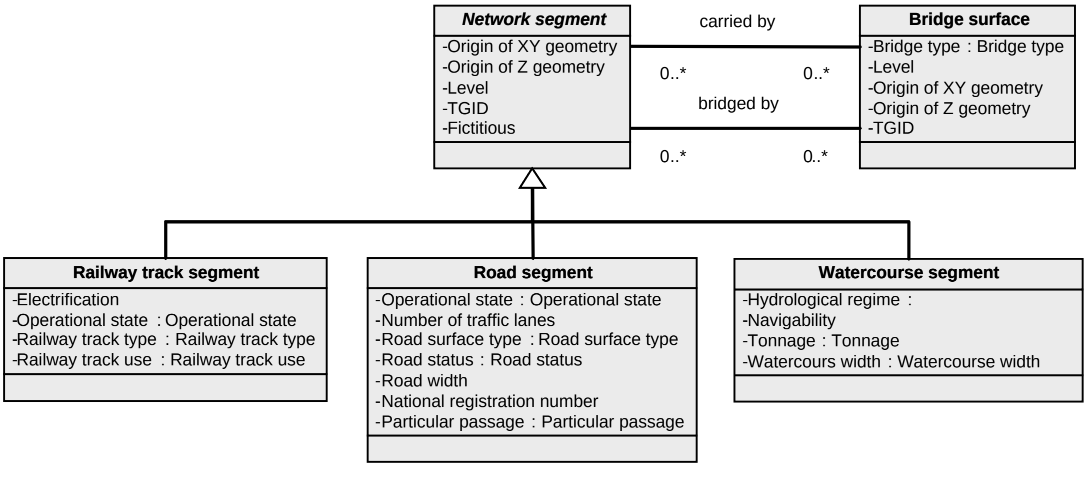
Network segment
-Origin of XY geometry
-Origin of Z geometry
-Level
-TGID
-Fictitious
carried by
0..*
0..*
bridged by
0..*
0..*
Bridge surface
-Bridge type : Bridge type
-Level
-Origin of XY geometry
-Origin of Z geometry
-TGID
Railway track segment
-Electrification
-Operational state : Operational state
-Railway track type : Railway track type
-Railway track use : Railway track use
Road segment
-Operational state : Operational state
-Number of traffic lanes
-Road surface type : Road surface type
-Road status : Road status
-Road width
-National registration number
-Particular passage : Particular passage
Watercourse segment
-Hydrological regime :
-Navigability
-Tonnage : Tonnage
-Watercours width : Watercourse width

Figure 1. Example of use of generalization relationships in the conceptual data model to move common attributes and associations from multiple descendant classes to a single ancestor class.

Generalization relations were used extensively to move common attributes and associations from multiple descendant classes to a single ancestor class. The use of these – often abstract – ancestors helped to simplify the model and indicate common structures, shared by feature types of different domains. A good example of this is the way in which railroads, roads and watercourses were all modeled by network segments which form an interconnected network (Fig. 1). Segments from the three types of networks all share the same types of relationships with bridges.

Logical modeling

The CDM was designed to be completely platform independent. In the next step of the data modeling process, a Logical Data Model (LDM) was derived from the CDM, adapting it to Esri's geodatabase model as proposed by Perencsik et al. (2004a). Esri provides a Visio template document (.vst) that contains all the basic Esri classes and interfaces. The Esri interfaces can be used to construct custom features with specific behavior. As no operations were defined for the feature types in the SGISR model, this possibility was not used and the focus was on the linking of the SGISR objects to the correct Esri basic classes. First the CDM was loaded into the Arcinfo UML model, organizing it into a number of different packages, based on the domains present in the model.

Next, the feature types were connected to the corresponding Esri classes via generalization relations. Non-geometrical types were coupled to the Esri object class while types with a geometrical component were coupled to the Esri feature class, inheriting its shape property and effectively identifying them as (geographical) features. For abstract types the IsAbstract property was checked to prevent them from being implemented later on as tables or feature classes in the database. Some feature types were linked to specialized Esri network features, such as the SimpleEdgeFeature and the SimpleJunctionFeatue, indicating their taking part in geometric networks.

Once the SGISR model had been successfully coupled to the basic Esri classes and the geometric networks were created, the organization of the feature types had to be adapted to the specific geodatabase model rules concerning the use of feature datasets. According to these rules all feature classes associated to a geometric network should be grouped inside the same feature dataset. Also, no object class can exist within a feature dataset. To adapt the model to these rules, some of the feature classes were moved to new packages who were stereotyped as feature datasets. This stereotyping indicates the fact that each of these packages corresponds to a dataset that is to be created during creation of the physical database schema.

The data types of the CDM attributes were replaced by the corresponding ESRI field types. The data types defining specific value domains were replaced by classes, stereotyped CodedValueDomain for coded value domains and stereotyped RangeDomain for range domains.

Finally, a number of tagged values were added to the model to provide additional information necessary for the successful creation of the database schema later on. Tagged values provide a way to set additional properties for UML elements (Perencsik et al. 2004b). They consist of a keyword and a value which are paired to add a custom property to any UML element. Table 1 shows the tagged values which were used in the SGISR LDM.

Table 1: Tagged values used in the SGISR Logical Data Model

Tagged value name	Values	Meaning
GeometryType	esriGeometryPoint esriGeometryPolygon esriGeometryPolyline	Sets the geometry type for feature classes
HasZ	True/False	Indicates whether a feature class contains Z-values
Precision	Integer	Sets the number of digits for integer or double type fields
Scale	Integer	Sets the number of decimal places in single and double type fields
Length	Integer	Sets the width of string type fields
AllowNulls	True/False	Indicates whether null values are allowed for a field
OriginClass	Class name	Defines the origin class for a relationship class
OriginPrimaryKey	Field name	Sets the primary key field of the origin class
OriginForeignKey	Field name	Sets the foreign key field of the origin class
DestinationPrimaryKey	Field name	Sets the primary key field of the destination class
DestinationForeignKey	Field name	Sets the foreign key field of the destination class

Creating the physical database schema

In the final phase of the data modeling process a physical database schema was created, based on the SGISR LDM that had been built in Visio.

First, the model stored in Visio had to be exported to a XMI (XML Metadata Interchange) file. The functionality to do this is not present in the standard Visio interface so an add-on called Esri XMI Export needed to be installed. The install process was carried out according to the guidelines found on Esri's support website (http://support.esri.com/geodatabase/uml). Two files were copied to the Visio install directory. The Esri XMIExport.vsl file is provided by Esri and can be downloaded from the Esri support website. The second file, XMIExprt.dll is a component provided by Microsoft and can be downloaded from the Microsoft download pages. Esri XMIExport.vsl was copied to "Visio install directory"\visio11\1033\ and XMIExprt.dll was copied to "Visio install directory"\visio11\DLL. To set up the add-on, some options needed to be set in Visio. In Tools/Options/Advanced/File paths, the directory where the Esri XMIExport.vsl file was located was added to the add-ons file paths. Although this completed the steps outlined in the guidelines, the export tool failed to appear in the Visio Add-On menu. Only after the directory where the XMIExprt.dll file had been placed had also been added to the add-ons file paths, did the add-on become available. It was subsequently used to export the SGISR model without any further problems.

The resulting XMI file containing the exported model was then checked for errors. The Arcinfo UML model template contains a macro called the Semantics Checker, which can be used to verify whether a model, contained in an XMI file, is completely compatible with the geodatabase model.

The macro uses a DLL file called UmlSemCheck.dll which can be found in the Arcgis installation folder. Before the macro could be executed, a reference had to be made to the UmlSemCheck.dll file in the Visual Basic Editor. The macro also uses a document type definition file called uml.dtd, which again can be found in the Arcgis installation folder. It defines the correct XMI document structure by providing a list of legal elements. For the Semantics Checker macro to work, the uml.dtd file had to be placed in the same directory as the XMI file containing the model.

A first try resulting in an error message revealed that an error in the macro's code prevented it from successfully accessing UmlSemCheck, although a reference to the DLL had been established. The original code of the macro consisted of these lines:

```
Sub Semantics_Checker()
   StartChecker
End Sub
```

It was adapted as follows to allow it to access UmlSemCheck:

```
Sub Semantics_Checker()
   UmlSemCheck.StartChecker
End Sub
```

The Semantics Checker was then run on the exported model. The resulting report contained a number of errors that were found in the model, documenting the names of the elements involved, the location of these elements within the model and a brief description of the problem. Most of these could be attributed to human errors in entering data in Visio and were easily resolved. One type of error however related directly to the data structure in the conceptual model. As previously stated, generalization relations were used extensively throughout the SGISR model to move shared properties and associations to common ancestors. Although the inheritance of attributes by classes from their ancestors is allowed by the Semantics Checker, the inheritance of associations is not, causing the following error message:

> *A feature class that has descendants is involved in a relationship class.*

As a consequence, all the inherited associations in the model had to be replaced by the individual associations at the descendant objects' level, adding a certain degree of – unnecessary – complexity to the LDM.

Once all corrections were made, the model was exported and checked again. From the corrected XMI a physical database schema was created in ArcCatalog, using the Schema Wizard tool. While this tool provides the opportunity to set a number of properties of the geodatabase, most of these had already been incorporated in the SGISR LDM using tagged values. Because of this, the interactive work during the schema creation was very limited. Only the spatial references of feature datasets and standalone feature classes were added to complete the model.

Some new errors that had not been detected by the Semantics Checker, appeared during the final steps of the Schema Wizard. It turned out that some of the names of feature attributes corresponded to reserved SQL keywords and had to be changed in the LDM. A more fundamental error occurred during the schema creation itself. A number of relationships were not created and the schema creation log file reported the following error message:

> *ERROR ---> Item not found in this collection.*

After analysis of the location of these errors in the model, it was found that they were caused by the fact that two feature classes took part in both an association and a composition relationship class as illustrated in Figure 2. In these cases only one of the relationship classes was created, the other one causing the error message in the log file. To prevent this and have the two relationship classes implemented, the composition relationship had to be changed to a normal association.

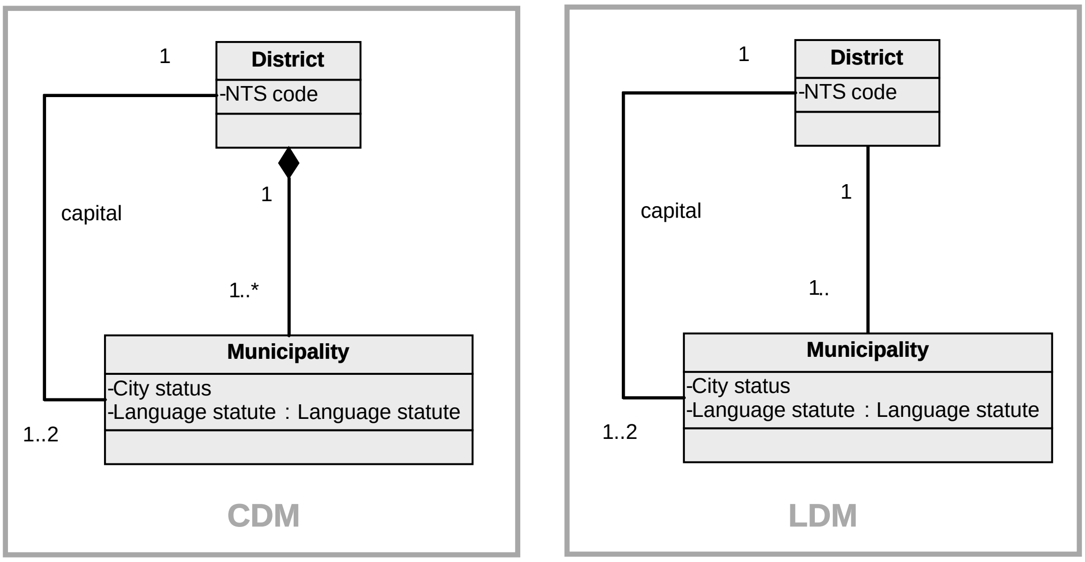

Figure 2. Two feature classes are linked by both an association and a composition relationship in the conceptual data model. In the logical data model, the composition had to be replaced by a second binary association.

Discussion and conclusions

Based on an analysis of the existing NGI vector data sets and the objectives of the SGISR project, a conceptual data model was created.

Having created the CDM in Visio, the SGISR project team opted to make use of a proposed workflow to adapt the conceptual model to the Esri geodatabase model, export this model to an XMI file and use this file to automatically create a physical database schema in ArcCatalog.

The functionality to complete this process is not fully integrated in the standard interfaces of the software applications involved. Additional software components needed to be installed and set up for both the XMI export tool and the Semantics Checker. Unfortunately, these procedures were not very well documented. In fact, some crucial steps were missing from the guidelines that were found on the Esri support website. This, and the error in the Semantics Checker macro, meant that neither of the tools functioned upon completion of the set up.

However, once the bugs had been fixed, both XMI export and Semantics Checker were easy to use and very helpful in adapting the SGISR model to comply with all the geodatabase rules. Together with the Schema Creation Wizard, they eliminated a lot of interactive work from the schema creation process.

In the end, the CDM was successfully adapted to the geodatabase model and a physical schema for the SGISR database was created.

Two data modeling issues were encountered during the process. These underline the difference between conceptual and logical data model and illustrate the usefulness of going through both stages. While some sacrifices had to be made in adapting to the chosen software platform, the platform independent vision of the model designer is preserved in the conceptual data model.

Acknowledgements

The authors would like to thank Jean-Charles Pruvost for his help during his internship at NGI Belgium. The continuing support of all the SGISR project team members is also greatly appreciated.

References

Anonymous. 2001. Draft International Standard ISO/DIS 19110, Geographic Information – Methodology for feature cataloguing. International Organization for Standardization. 53p.

Perencsik, A., Idolyantes, E., Booth, B., and Andrade, J. 2004. Designing Geodatabases with Visio. ESRI. 60p.

Perencsik, A., Idolyantes, E., Booth, B., and Andrade, J. 2004. Introduction to CASE Tools. ESRI. 12p.