Generating XML/GML Schemas from Geographic Conceptual Schemas

André C. Hora, Clodoveu A. Davis Jr., and Mirella M. Moro

Department of Computer Science
Federal University of Minas Gerais
Belo Horizonte, Brazil
{andrech,clodoveu,mirella}@dcc.ufmg.br

Abstract. A large volume of data with complex structures is currently represented in GML (Geography Markup Language) for storing and exchanging geographic information. As the size and complexity of such documents and their schemas grow, techniques and rules for designing and creating such documents become indispensable. This paper introduces a method for mapping geographic conceptual specifications (defined in OMT-G) to GML Schema. Our method avoids semantic or structural losses and provides redundancy-free data. It also reduces the use of integrity constraints and improves the nesting of XML elements in the resulting schema. We have implemented the method in order to automate the process of obtaining the target schema from the original geographic model. Experimental results show that spatial and non-spatial queries over the GML documents created from schemas generated using our method are more efficient than on documents created with a traditional, direct mapping process.

1 Introduction

Conceptual design is a decisive step for the successful implementation of geographic applications. If done correctly, the conceptual design allows the early detection of flaws, the adequate selection of representation alternatives, and a vision of the system requirements as to the most important transactions. There is a consensus around the need for a correct conceptual design as an early step in the design of database-centric applications, and geographic applications are no exception.

Considering that need, some conceptual models for geographic applications have been proposed. The Object Modeling Technique for Geographic applications (OMT-G) [1] is one of such models. It extends Unified Modeling Language (UML) concepts and diagrams to include geographic representations and special kinds of relationships. It also provides tools for the design of transformations over the basic data representation and specify visualization requirements for geographic data. The OMT-G model has been developed by our group and is currently used by GIS developers in many governmental, industrial and academic organizations in Brazil.

On a different perspective, XML is being widely adopted as a standard language for data representation, exchange and storage. Its auto-descriptive structure and its textual and non-proprietary format (which facilitates the creation of documents both by humans and by software) are among the reasons behind this widespread adoption. The

existence of several additional languages for supporting and manipulating XML documents, such as XPath, XQuery, XML Schema, DTD and Relax NG, makes its use even more attractive for data management.

The most important XML variation dedicated to geographic data and applications, the Geography Markup Language (GML[1]), is a standard developed and promoted by the Open Geospatial Consortium (OGC[2]), along with GML Schema. GML Schema is a geographic extension of XML Schema, the schema definition language standard for XML data. In many situations, users are expected to directly encode data and schemas in GML and GML Schema, including in the configuration of WebGIS packages. Data stored in geo-enabled DBMSs, such as Oracle or PostGIS, can be exported to or imported from XML or GML. GML documents are also used in the specification of OGC Web services such as Web Map Service (WMS[3]), Web Feature Service (WFS[4]) and others, which are important resources for establishing spatial data infrastructures. However, given the usual complexity and the peculiarities of geographic applications, creating a database structure directly in GML Schema is not easy. Furthermore, the interaction between application designer and specialist user, a fundamental task in database applications design, is much harder to achieve. It would be more natural to design using a conceptual model, taking advantage of the visual nature of class diagrams and other visual tools for the interaction with the specialist user, before trying to encode the database structure in GML.

This paper presents a methodology for mapping OMT-G conceptual schemas into GML Schema. The mapping avoids semantic or structural losses, preserving the information encoded in the conceptual design (such as representation choices and spatial relationships), while reducing the use of integrity constraints and improving the nesting of XML elements in the resulting schema. This methodology is part of a larger effort by our group, which will ultimately lead to the development of a computer-aided geographic applications design tool for all the required steps between conceptual modeling and physical implementation of geographic databases.

This paper is organized as follows. Section 2 presents related work. Section 3 briefly describes the OMT-G model. Section 4 presents the method for transforming OMT-G schemas to GML Schema, including mapping rules and a transformation algorithm. Section 5 shows a case study and some experimental results. Finally, Section 6 shows our conclusions and outlines future work.

2 Related Work

Techniques have been proposed in the literature to perform the mapping of ER (entity-relationship) or EER (extended entity-relationship) schemas to XML schemas. Pigozzo and Quintarelli [2] present an algorithm to generate XML schemas from EER. The resulting schema contains all the characteristics of the original one: entities, relationships, attributes, cardinality and specializations. Primary and foreign keys are used in order to

[1] Geography Markup Language (GML): http://www.opengeospatial.org/standards/gml
[2] Open Geospatial Consortium (OGC): http://www.opengeospatial.org
[3] Web Map Service (WMS): http://www.opengeospatial.org/standards/wms
[4] Web Feature Service (WFS): http://www.opengeospatial.org/standards/wfs

avoid redundancy in the resulting schema. Some EER entities are selected, considering the cardinality of their relationships, to be mapped to root elements in XML Schema, therefore becoming *first-level entities*. The mapping algorithm also uses *inclusion conditions*, in which some constraints are established as to the insertion of sub-elements in the XML schema. The resulting schema follows the XML Normal Form [3].

Franceschet et al. [4] show the mapping of the spatio-temporal model ChronoGeoGraph into XML/GML schemas. In [5] a mapping from EER to XML Schema is proposed, with the following properties: preservation of modeling information and integrity constraints, absence of redundancy, hierarchical views of the conceptual information, highly connected structure, and reversible result (i.e., from the XML schema it is possible to obtain the EER schema). The mapping occurs first from EER to an intermediate language, called XLS. The transformation from XLS to XML Schema is performed in a direct fashion, and is defined through steps that detail the mapping of entities, relationships, weak entities and specializations. However, the work does not explain which entities of the EER schema are selected to become first-level entities, nor how the mapping of attributes is executed.

Liu and Li [6] propose the creation of XML schemas from ER. Some quality criteria for the creation of XML schemas are discussed, including the preservation of information, absence of redundancy, strongly nested structure, and reversible result. A recursive transformation algorithm that follows these quality criteria is then presented. The algorithm is based on twelve rules that map entities, relationships, generalizations and attributes to XML Schema. As in the previous work, the criteria for the selection of first-level entities are not made clear.

There is also some work that uses different conceptual models to generate XML schemas. Al-Kamha et al. [7] focus the representation of generalization and specialization in XML schemas using the Conceptual XML (C-XML) model. The techniques they present are capable of representing the inheritance relationships with a reduction in the complexity of the resulting XML schema when it involves generalization or specialization. Mok and Embley [8] present the creation of XML structures that are optimized by analyzing the constraints in a generic conceptual model. Some commercial tools provided by IBM [9, 10] and Sparx [11] include software or libraries to convert a conceptual schema into an XML schema. In order to simplify the transition between conceptual and XML schemas, several approaches have been proposed, and a comparison among the main ones is shown in [12]. Going in a slightly different direction, Kleiner and Lipeck [13] present an algorithm for the generation of DTD schemas from ER schemas.

Most of the discussed work uses XML Schema as the base for the generation of schemas, due to its greater expressiveness in comparison to other alternatives. There is a strong concern with the preservation of the semantics contained in the original schema, so that no information is lost in the transformation. Other common goals of the techniques we reviewed were the absence of redundancy, the use of nested structures, and reversibility. Only one source discusses specificities of geographic data models and applications and their transformation to XML/GML [4].

3 The OMT-G Model

OMT-G uses the primitives defined for the UML class diagram and introduces geographic characteristics in order to enhance its capacity of representing the semantics of spatial data. OMT-G includes primitives to model the geometry and the topology of geographic data, thus supporting structures such as spatial aggregation, arc-node networks, topological relationships and multiple representations [14].

Classes and relationships are the basic primitives for the creation of OMT-G schemas. Classes can be either *conventional* or *spatial*. Conventional classes do not have geographic properties and behave exactly as UML classes. Spatial classes include a geographic representation, which can be individualized (geo-object) or space-covering (geo-field). Geo-objects can be represented using points, lines, polygons or network elements (nodes, unidirectional arcs and bidirectional arcs). Geo-fields represent continuously varying phenomena, usually seen as surfaces, and their geometric representation includes isolines, tessellation, planar subdivision, triangulation, or sampling. Relationships can also be conventional (equivalent to UML relationships) or spatial. Spatial relationships include topological relationships (disjoint, covers, contains, and so on), network relationships and spatial aggregations ("whole-part"aggregations). Generalizations and specializations are specified like in UML, and they can be total/partial or disjoint/overlapping. Generalizations and specializations require participating classes to have the same type of representation. Another primitive, called conceptual generalization, allows the modeling of objects with multiple geographic representations. We refer to [1, 14, 15] for more information on OMT-G.

Figure 1 presents an OMT-G schema for a simple urban geographic application. Classes *region* and *neighborhood* are represented as geo-fields, since they cover the entire space of interest with a set of adjacent polygons, forming a planar subdivision, a type of representation commonly used for political boundaries. Classes *block*, *building*, *public building*, *private building*, *street crossing* and *street segment* are represented using geo-objects. *Thoroughfare* is a conventional class, i.e., it does not have a geometric representation or a location associated to their members. Spatial relationships are depicted in dashed lines, while non-spatial ones in solid lines. The relationship between street crossing and street segment forms an arc-node network. A total/disjoint specialization relationship exists in which *building* is the superclass.

4 Mapping OMT-G schemas to XML/GML Schemas

After presenting the basics for OMT-G, the subsections describe the rules for mapping OMT-G schemas to XML schemas and the mapping algorithm.

4.1 Mapping Rules

The transformation of the conceptual schema, which is built as a graph, into a GML Schema, which uses a hierarchy, is not a direct process [16]. Nevertheless, the mapping must be made so that semantics and structure are preserved. Here we propose a set

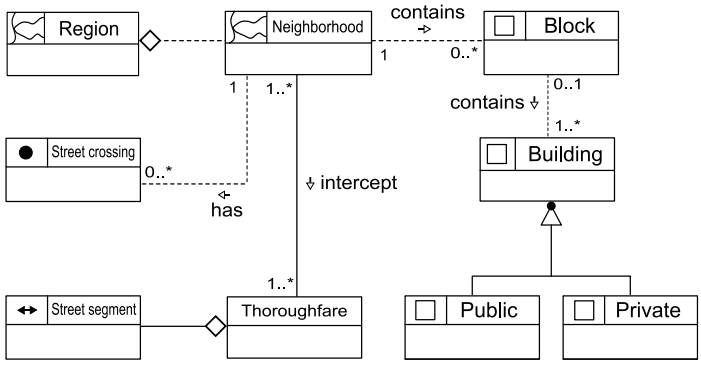


Fig. 1. OMT-G schema for a simple urban geographic application

of mapping rules that fulfill such requirements. A complete example using the rules is illustrated in Section 5.

OMT-G Class Diagram: a root element is created in the GML schema to represent the OMT-G class diagram from the original schema. All elements mapped to the target schema are inserted as sub-elements of the OMT-G class diagram.

Spatial Domain: represents the spatial range of the elements in the GML schema. All elements of a GML document based on the resulting schema are topologically contained in this range. In the target schema, the domain appears as an optional sub-element of the root through the GML Schema geometrical properties[5] `boundedBy` (which describes the approximate shape of the spatial domain, such as a bounding rectangle) and `extentOf` (which describes the actual shape of the spatial domain). OMT-G schemas do not include this information explicitly, so it must be provided by the user when creating documents from the GML schema.

Conventional and Spatial Entity: OMT-G conventional classes are mapped to XML elements. Spatial classes are also mapped to elements, but they also receive a sub-element corresponding to the representation type, as defined in the OMT-G schema. Classes represented in OMT-G as Point, Node or Sample are mapped to the GML `PointType`; Line, Unidirectional Arc and Bidirectional Arc are mapped to GML `LineStringType`; OMT-G Polygons and Planar Subdivisions are mapped to GML `PolygonType`. Isolines are mapped to `LineStringType` and/or `PolygonType`, and triangulations are mapped to `PointType` (triangle vertices) and `PolygonType` (triangles). Tesselations are mapped to regular grids, encoded as GML grid coverages, but we will not detail this mapping further, due to space limitations.

Relationships: in spatial databases, spatial relationships are verified on-the-fly, using topological functions. Thus, OMT-G spatial relationships indicate spatial integrity constraints, rather than materialize connections between objects. Since GML Schema does not represent topological constraints, spatial relationships are mapped to conventional ones. Relationships are then mapped using a strategy designed to reduce the usage of XML constraints `key` and `keyref` by nesting elements as much as possible, thus

[5] Geometric properties in GML are used to describe the role of the geometry in a relation [17]

contributing to improve the performance of query execution over derived XML/GML documents. This mapping is done according to the rules indicated in Table 1. The header row and column in Table 1 indicate, respectively, the cardinality of the participation of generic classes `A` and `B` in the relationship `R`. Cardinality is shown in *(min, max)* notation [18] and is inverted as to what is read in the OMT-G class diagram, i.e., the cardinality indicated in each row header indicates the participation of instances from `A` in the relationship, which is indicated at the `B` side of the relationship in OMT-G, and vice-versa. In the cells of Table 1, the mapping rules, based on [5], are shown using the following notation: `A(kA, R[min,max])` indicates an element `A` and its sub-elements. `R[min,max]` indicates a relationship in which `A` participates, with `min` to `max` cardinality, and the key attribute for the element `A` is indicated as `kA` (additional attributes are omitted for simplicity). Nesting of elements is indicated using indentation. Keys for referential integrity constraints are denoted using `key` and `keyref`.

As an example, consider the mapping of the relationship between *neighborhood* and *block* (Figure 1). We see that each neighborhood contains zero or more blocks, and that blocks have exactly one corresponding neighborhood. The mapping rules for this case are indicated in the cell at the second row and third column of Table 1 (i.e., `A(0,n)` corresponds to neighborhood, and `B(1,1)` corresponds to block). An element will be created for each neighborhood, containing its key and the *contains* relationship to blocks; since there can be zero or more blocks inside each neighborhood, the minimum and maximum number of occurrences of *contains* is `(0,n)`. Block elements are then nested into each neighborhood, within sub-elements for each instance of the relationship. Block elements are then defined, nested within their corresponding neighborhoods through the *contains* relationship. Key elements for neighborhoods and blocks are then defined. Observe that, in this case, no keyrefs were needed.

Network relationships require the association of each arc to exactly two nodes, while each node can be related to one or more arcs. This constitutes a special case of the rules in Table 1, which is resolved in a similar fashion: (1) an element is created for the arcs class, containing its attributes and the relationship with `(2,2)` cardinality; (2) two sub-elements, corresponding to the relationship of the arc to each related node, are created, using references to the nodes' keys; (3) elements are created for each node; (4) keys for the arcs and for the nodes are created; (5) two keyrefs are created for each arc, one for each arc-node association. Aggregations are mapped exactly as `(1,1)` to `(1,n)` conventional relationships.

Generalization and Conceptual Generalization: generalizations and specializations in OMT-G can be either total or partial, and can be either disjoint or overlapping. An XML Schema element for sequence or choice is used in the mapping of the superclass (depending on the type of generalization in the source schema) to insert elements corresponding to the subclasses. If the generalization is total and disjoint, the element `choice` is used, since only one (exactly one) of the subclasses can occur. If it is total and overlapping, the constraints `minOccurs=1` and `maxOccurs=`*number of subclasses* are used, since any non-empty combination is allowed. The partial-disjoint relationship also uses `choice`, but establishes `minOccurs=0` and `maxOccurs=1`, since a superclass element can either belong to a subclass, or to none. Finally, the partial-overlapping relationship uses the element `sequence`, and enforces the constraints

Table 1. Relationships mapping

A/B	(0,1)	(0,n)	(1,1)	(1,n)
(0,1)	A(kA, R[0,1]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *key*(R.kB) *keyref*(R.kB→B.kB)	A(kA, R[0,1]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *keyref*(R.kB→B.kB)	A(kA, R[0,1]) R(B) B(kB) *key*(A.kA), *key*(B.kB)	A(kA) B(kB, R[1,n]) R(kA) *key*(A.kA), *key*(B.kB) *key*(R.kA) *keyref*(R.kA→A.kA)
(0,n)	A(kA) B(kB, R[0,1]) R(kA) *key*(A.kA), *key*(B.kB) *keyref*(R.kA→A.kA)	A(kA, R[0,n]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *keyref*(R.kB→B.kB)	A(kA, R[0,n]) R(B) B(kB) *key*(A.kA), *key*(B.kB)	A(kA) B(kB, R[1,n]) R(kA) *key*(A.kA), *key*(B.kB) *keyref*(R.kA→A.kA)
(1,1)	B(kB, R[0,1]) R(A) A(kA) *key*(B.kB), *key*(A.kA)	B(kB, R[0,n]) R(A) A(kA) *key*(B.kB), *key*(A.kA)	A(kA, R[1,1]) R(B) B(kB) *key*(A.kA), *key*(B.kB)	B(kB, R[1,n]) R(A) A(kA) *key*(B.kB), *key*(A.kA)
(1,n)	A(kA, R[1,n]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *key*(R.kB) *keyref*(R.kB→B.kB)	A(kA, R[1,n]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *keyref*(R.kB→B.KB)	A(kA, R[1,n]) R(B) B(kB) *key*(A.kA), *key*(B.kB)	A(kA, R[1,n]) R(kB) B(kB) *key*(A.kA), *key*(B.kB) *keyref*(R.kB→B.kB)

`minOccurs=0` and `maxOccurs=1` on each subclass, since any combination of subclasses is allowed. Conceptual generalization in OMT-G can be either according to geometric shape or according to scale. In both cases an XML Schema element for sequence is used in the mapping of the superclass to insert elements corresponding to the subclasses geometry type, and enforces constraints `minOccurs=0` and `maxOccurs=1` on each subclass, since any combination of subclasses is allowed.

Attributes and Constraints: OMT-G attributes are transformed into single-valued or multi-valued properties, and include primary key, foreign key, width and domain constraints. Primary keys of classes are mapped into elements, and primary keys of relationships are mapped to attributes, both with the `key` constraint. Foreign keys are created using the `keyref` constraint, according to the relationships defined for the class. Single-valued attributes are mapped to elements using XML primitive types. Multi-valued attributes use the constraint elements `minOccurs` and `maxOccurs` to represent their cardinality. Attributes which have size constraints use the `minLength` and `maxLength` elements, and those subject to domain constraints use the `enumeration` element.

4.2 Mapping Algorithm

We propose an extension of the mapping algorithm presented in [2] to encompass the needs of the OMT-G to GML Schema mapping. The mapping consists of two phases: the first one determines the first-level elements, i.e., the classes that will be encoded directly as root elements, and the second generates the GML schema itself.

First-level elements are initially derived from classes that fulfill some requirements as to the relationships in which they participate. At least one of the following conditions must be met for a class to be mapped to a first-level element: (1) it participates partially in at least one relationship, (2) it participates totally in at least one many-to-many or

Algorithm 1 Mapping OMT-G to GML Schema

Input: an OMT-G schema
Output: the mapped GML Schema

Create the OMT-G diagram *root element*
Add the spatial domain to the root element as a *sub-element*
Define the first-level elements
for each class c that is not mapped to GML Schema (starting from first-level elements) **do**
    MapClass(c)
end for

procedure MapClass(c: a class c in OMT-G Schema)
Add c to the current element as a *sub-element*
Add an id to the c element as a *sub-element*
Add $c.id$ to the root element as a *key*
Add the geometry and non-spatial attributes of c to the c element as *sub-elements*
for each relationship, generalization or arc-node network r involving c that is not already mapped to GML Schema **do**
    Let $c1$ be the class corresponding to c and $c2$ be the related class in r
    if (the cardinality of r is one-to-one) **then**
        if ((the participation of $c1$ is partial and $c2$ is total) **or** (the participation of $c1$ and $c2$ is partial) **or** (the participation of $c1$ and $c2$ is total **and** *InclusionCondition*($c2$) holds)) **then**
            Add r to the $c1$ element as a *sub-element*
            MapKeyConstraint(r, $c2$) // process key and keyref of r
        end if
    else if (the cardinality of r is one-to-many) **then**
        if (($c1$ is at the "one" side **and** (*InclusionCondition*($c2$) holds **or** the participation of $c1$ is total)) **or** ($c1$ is at the "many" side **and** the participation of $c1$ and $c2$ is partial)) **then**
            Add r to the $c1$ element as a *sub-element*
            MapKeyConstraint(r, $c2$)
        end if
    else if (the cardinality of r is many-to-many) **then**
        if ((the participation of $c1$ and $c2$ is partial) **or** (the participation of $c1$ and $c2$ is total) **or** (the participation of $c1$ is total and $c2$ is partial)) **then**
            Add r to the $c1$ element as a *sub-element*
            MapKeyConstraint(r, $c2$)
        end if
    else if (r is a generalization) **then**
        Add r to the $c1$ element as a *sub-element*
    else if (r is an arc-node network **and** $c1$ is the arc) **then**
        Add relationships $r1$ and $r2$ to the $c1$ element as a *sub-element*
        Add $r1.id$ and $r2.id$ to the root element as a *keyref* referencing $c2.id$
    end if
    if ($c2$ has already been translated) **then**
        Add r to the $c1$ element as a *sub-element*
        Add $r.id$ to the root element as a *keyref* referencing $c2.id$
    end if
    if ((the cardinality of r is one-to-one or one-to-many **or** r is a generalization) **and** *InclusionCondition*($c2$) holds) **then**
        MapClass($c2$) // recursively calls
    end if
end for

procedure MapKeyConstraint(r: a relationship, c: a class)
Let n be the cardinality of c in r; min be minimum and max be maximum value of n; id be an identifier of r
Add $r.id$ to the r element as an *attribute*
if ($min = 0$ **and** $max = 1$) **then**
    Add $r.id$ to the root element as a *key*
end if
if ($min \neq 1$ **and** $max \neq 1$) **then**
    Add $r.id$ to the root element as a *keyref* referencing $c.id$
end if

function InclusionCondition(c: a class) // return true if the conditions hold
return c has not been translated yet **and** the participation of c is total **and** c is not a subclass **and** c is not a part class **and** c is not a first-level element

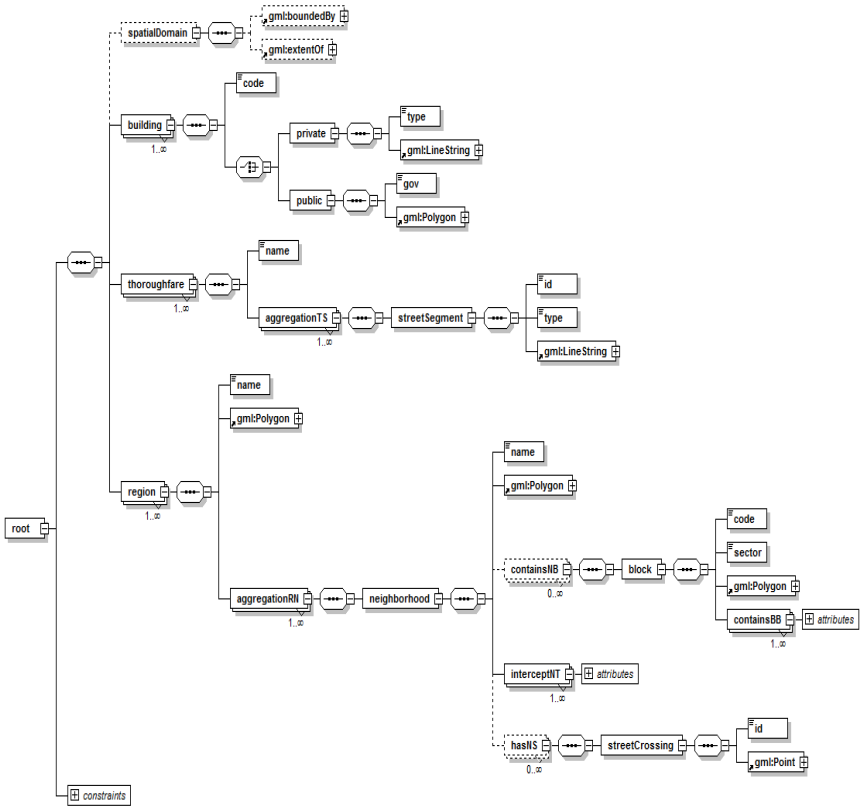


Fig. 2. Hierarchical tree of the schema generated by the proposed approach

one one-to-many relationship, (3) it is the superclass in a generalization, but it does not participate in any other relationships, (4) it is the "whole" class in an aggregation, and it does not participate in any other relationships. These conditions are sufficient, but not necessary, for an element to be treated in the first level, i.e., other classes may be mapped to the first level as well later on the mapping, depending on the source schema.

Algorithm 1 shows the pseudocode for the recursive mapping algorithm, based on the rules presented in section 4.1. In line 1 the root element of the OMT-G class diagram is created. In line 2 the spatial domain is defined, and in line 3 the first-level elements are created. The loop between lines 4 and 6 calls the `MapClass` procedure (lines 7-42) to map each OMT-G class. Lines 8-11 add the class as a sub-element of the current element, along with its identifier (primary key) and its spatial and non-spatial attributes, as defined by the attribute rules on section 4.1. The loop between lines 12 and 42 maps each relationship (depending on its cardinality), generalization and arc-node network, and also takes care of the primary and foreign keys (`key` and `keyref`) corresponding to these relationships using the `MapKeyConstraint` procedure (lines 43-51). Line 40 recursively calls the `MapClass` procedure, allowing nested structures to be created in the XML/GML schema.

5 Case Study and Experimental Results

We verified the correctness of the mapping using a case study. We also used the case study schema and data to verify the performance gains that can be achieved by its use, as compared to a naive and direct mapping. The nesting strategy used in the algorithm was expected to generate performance gains due to the reduction of the number of access to references (i.e., primary keys and foreign keys) and of the number of joins (replaced by hierarchical accesses in the nested structure). Some spatial extensions for geographic querying GML documents have been proposed [19, 20, 21]. We used XQuery and GQL [20], an XQuery extension that supports spatial operators through Oracle's XQLPlus tool. Algorithm 1 has been implemented in Java using the JDOM library.

The OMT-G schema presented in Figure 1 has been mapped to two XML/GML schemas: a flat one and a nested one. The latter has been mapped using the proposed algorithm. Figure 2 shows the structure of the hierarchical tree created by the algorithm. Notice that *building*, *thoroughfare* and *region* were created as first-level elements, while the others were nested into them according to their relationships.

An instance of the OMT-G schema in Figure 1 was loaded to the Oracle database and used to create two XML/GML documents. This instance is composed of: 2 *regions*, 83 *neighborhoods*, 2.215 *blocks*, 2.638 *thoroughfares*, 12.078 *streetSegments*, 10.167 *buildings* and 5.365 *streetCrossings*. Each document was created based on each XML/GML schema (flat and nested), using functions from Oracle's XML DB and Spatial products. Ten queries were then executed over each document in order to verify the resulting performance. Table 2 shows the description of the queries. Queries 1 to 5 are non-spatial and have been specified in XQuery, while queries 6-10 are spatial and have been written using GQL[6].

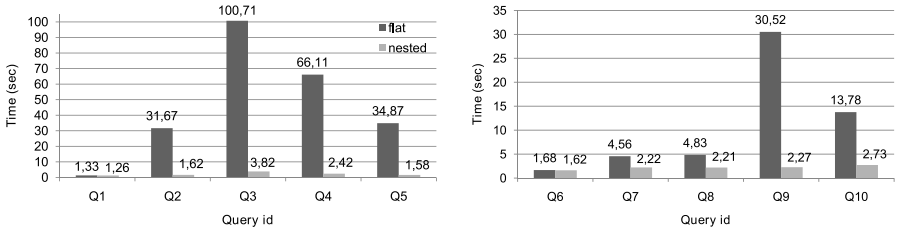
Table 2. Queries used in the case study

Q1	Select the neighborhoods of "Centro-Sul" region	Q6	Select the neighborhoods area of "Centro-Sul" region
Q2	Select the blocks of "Oeste" region	Q7	Select the neighborhoods blocks with distance < 50 of "Barroca" or "Alto Barroca" neighborhoods
Q3	Select the street segments of "Centro" neighborhood	Q8	Select the neighborhoods blocks that touch "Gutierrez" or "Centro" neighborhoods
Q4	Select the street segments amount of the thoroughfares that intersect "Funcionários" neighborhood	Q9	Select the street segments length of the thoroughfares that intersect "Centro" neighborhood
Q5	Select the blocks of "Centro-Sul" region with sector < 10	Q10	Select the neighborhoods street crossings that touch "Sion", "Santo Agostinho", "Salgado Filho" or "Estoril" neighborhoods

The experiments were performed in a 2 GHz Intel Core2Duo computer with 3 GB of RAM. Results are presented in Figure 3, which shows the running time for queries performed over each document. In the experiments, the nested approach resulted in much better query times for both types of queries (for instance, Q3 results show a difference of two orders of magnitude between the nested and the flat documents). Although we

[6] The schemas, documents and queries used in this paper are available at http://www.dcc.ufmg.br/~andrech/omtg2xml

certainly cannot generalize this result for every schema or document, results show that nested documents tend to perform much better than flat ones, thus justifying our mapping approach.



(a) non-spatial queries (b) spatial queries

Fig. 3. Results of the queries executed in the experiment

6 Conclusion

This paper introduced a mapping technique for conceptual schemas from geographic applications to GML Schema. Such technique allows developers that have Web applications as their final goal to start with higher-level conceptual design tools. The paper also showed, through a performance comparison between flat and nested versions of a GML document, that the mapping algorithm can lead to better results in practical use of the data encoded in GML.

Future work includes the final specification and implementation of the mapping of tessellations as well as the development of a graphical user interface for OMT-G conceptual modeling. We intend the tool to be able to (1) allow interactive design of OMT-G diagram, (2) execute the mapping to GML Schema, (3) reverse the mapping, generating OMT-G diagrams from given GML Schemas, and (4) include other mapping levels to allow the physical implementation to take place in XML/GML or in a spatially-extended object-relational geographic DBMS.

Acknowledgments

The authors acknowledge the support from FAPEMIG and CNPq, Brazilian agencies in charge of fostering research and development, in the form of individual research grants and scholarships, and also the support from the Brazilian National Institute of Science and Technology for the Web (CNPq grant 573871/2008-6).

References

[1] Borges, K.A.V., Davis, C.A., Laender, A.H.F.: OMT-G: An Object-Oriented Data Model for Geographic Applications. Geoinformatica **5**(3) (2001) 221–260

[2] Pigozzo, P., Quintarelli, E.: An Algorithm for Generating XML Schemas from ER Schemas. In: Proceedings of Italian Symposium on Advanced Database Systems - SEBD. (2005) 192–199
[3] Arenas, M., Libkin, L.: A Normal Form for XML Documents. In: Proceedings of Symposium on Principles of Database Systems - PODS. (2002) 85–96
[4] Franceschet, M., Montanari, A., Gubiani, D.: Modeling and Validating Spatio-Temporal Conceptual Schemas in XML Schema. In: Proceedings of International Conference on Database and Expert Systems Applications Workshops - DEXA. (2007) 25–29
[5] Franceschet, M., Gubiani, D., Montanari, A., Piazza, C.: From Entity Relationship to XML Schema: A Graph-Theoretic Approach. In: Proceedings of International XML Database Symposium - XSym. (2009) 165–179
[6] Liu, C., Li, J.: Designing Quality XML Schemas from E-R Diagrams. In: Proceedings of International Conference on Web-Age Information Management - WAIM. (2006) 508–519
[7] Al-Kamha, R., Embley, D.W., Liddle, S.W.: Representing Generalization/Specialization in XML Schema. In: Proceedings of Enterprise Modelling and Information Systems Architectures - EMISA. Volume 75. (2005) 93–104
[8] Mok, W.Y., Embley, D.W.: Generating Compact Redundancy-Free XML Documents from Conceptual-Model Hypergraphs. IEEE Trans. on Knowl. and Data Eng. **18**(8) (2006) 1082–1096
[9] IBM Alphaworks: Model Transformation Framework. http://www.alphaworks.ibm.com/tech/mtf (2009) last access on January 2010.
[10] IBM Rational Software: Generating XSD Schemas from UML Models. http://publib.boulder.ibm.com/infocenter/rtnlhelp/v6r0m0/topic/com.ibm.xtools.transformations.doc/topics/txsdover.html (2009) last access on January 2010.
[11] Sparx Systems: Sparx Systems UML to XML Schema Transformation. http://www.sparxsystems.com/resources/mda/xsd_transformation.html (2009) last access on January 2010.
[12] Necasky, M.: Conceptual Modeling for XML: A Survey. In: Proceedings of International Workshop on Databases, Texts, Specifications and Objects - DATESO. Volume 176. (2006) 40–53
[13] Kleiner, C., Lipeck, U.W.: Automatic Generation of XML DTDs from Conceptual Database Schemas. In: Proceedings of Wirtschaft und Wissenschaft in der Network Economy Tagungsband der GI/OCG Jahrestagung - Informatik. (2001) 396–405
[14] Borges, K.A.V., Davis, C.A., Laender, A.H.F.: Integrity Constraints in Spatial Databases. In: Proceeding of Database Integrity: Challenges and Solutions. (2002) 144–171
[15] Borges, K.A.V., Davis, C.A., Laender, A.H.F.: Modelagem conceitual de dados geográficos. In Casanova, M., Câmara, G., Davis, C.A., Vinhas, L., Ribeiro, G., eds.: Bancos de Dados Geográficos. MundoGeo Editora, Curitiba (2005) 83–136
[16] Schroeder, R., Mello, R.D.S.: Designing XML Documents from Conceptual Schemas and Workload Information. Multimedia Tools Appl. **43**(3) (2009) 303–326
[17] Lake, R., Burggraf, D., Trninic, M., Rae, L.: Geography Mark-Up Language: Foundation for the Geo-Web. John Wiley & Sons (2004)
[18] Elmasri, R., Navathe, S.B.: Fundamentals of Database Systems (5th Edition). Addison Wesley (March 2006)
[19] Boucelma, O., Colonna, F.M.: Querying GML data with GQuery. Technical report, University of Provence (2004)
[20] Guan, J., Zhu, F., Zhou, J., Niu, L.: GQL: Extending XQuery to Query GML Documents. In: Proceedings of Geo-Spatial Information Science. Volume 9. (2006) 118–126
[21] Vatsavai, R.R.: GML-QL: A Spatial Query Language Specification for GML. Technical report, University of Minnesota (2002)