Mapping spatio-temporal conceptual schemas into XML Schema documents

Massimo Franceschet and Angelo Montanari
Department of Mathematics and Computer Science
University of Udine
Udine - Italy

Donatella Gubiani
Department of Science
University G. D'Annunzio of Chieti-Pescara
Pescara - Italy

Abstract

In this paper, we describe a translation algorithm that maps spatio-temporal conceptual schemas into XML schemas expressed in the W3C XML Schema Language. Moreover, we propose a suitable Java library to validate XML documents with respect to the translated schemas.

1. Introduction

Geographic information which evolves over time is involved in several database applications, including many environmental, social, and economic systems [**?**]. Spatio-temporal conceptual models can be successfully exploited in the development of these applications to provide the setting for the interaction between customers and developers. In particular, they allow the developers to detect possible inconsistencies and incompleteness in customers' requirements. Moreover, since conceptual schemas are independent of any specific implementation, they can be used in the case of technology upgrade and transfer. We focus our attention on the ChronoGeoGraph (CGG for short) formalism for conceptual modeling of spatio-temporal databases [6]. CGG extends the Entity-Relationship (ER) formalism with a number of constructs that capture spatio-temporal aspects of the application domain. In particular, CGG encompasses multiple temporal dimensions, ranging from the classical dimensions of transaction and valid times to the recently-proposed event and availability time dimensions [2], it supports both the object-based and the field-based view of spatial information, and it makes it possible to describe the temporal evolution of geometrical properties, such as location and shape, of the modeled entities/phenomena, e.g., a contaminated site or a storm.

The contribution of the present work is twofold:

1. we propose a mapping from CGG conceptual schemas into W3C XML Schema Language schemas [17, 18] (XML schemas, in the following) and implement it in the Java programming language;
2. we design and implement a Java validation library for spatio-temporal constrains.

The CGG project [**?**] includes a tool for the visual synthesis of CGG conceptual schemas. This tool has been extended with the translation algorithm that maps CGG schemas into XML ones. The validation library is used to validate XML documents with respect to the generated XML schemas. All in all, we developed a tool to validate spatio-temporal information, stored in XML documents, with respect to spatio-temporal constrains, expressed as visual CGG conceptual schemas.

The proposed mapping has the following advantages:

1. Since an XML schema is an XML document, one can take advantage of XML query languages, such as XQuery [19], to query the XML schema. For instance, one might want to detect structural properties of the original schema, e.g., to check whether there exists a path which satisfies suitable conditions, that connects a specific pair of entities of the conceptual schema (reachability problem). As an example, consider the case of a conceptual schema that represents the contaminated sites of a region [3]. Such a schema includes an entity commune and an entity site linked together by either an inclusion relation (the communal territory includes the site) or an overlapping one (communal territory and site overlap). To take into account the possibility that contamination spreads by the rivers, one may extend the schema with an entity river linked to both the entity commune and the entity site by a (distinct) cross relation (a river may cross a communal territory as well as a contaminated site). Once a conceptual schema is mapped into an XML one, one can query the XML schema to determine all paths connecting any pair of entities. In the considered case, such a capability can be exploited to compute the set of all direct and indirect paths that link the entity site to the entity commune, which includes the path passing through the entity river.

2. Since both the schema and its instance are stored as XML documents, one can query both the schema and the instance in the same query. For instance, one might want to retrieve all key values of spatial entities in the database. Entity key values are stored in the XML instance whereas the fact that an entity is a spatial one is said in the XML schema.

3. The mapping of CGG schemas into XML ones also supports the exchange and integration of spatio-temporal data among different applications. In particular, it allows one to take advantage of definitions of the standard Geography Markup Language (GML), proposed by the OpenGIS Consortium (OGM) [14], which is based on XML Schema. XML, and in particular GML in the spatial context, is indeed becoming the standard for publishing and exchanging data on the web. On the one hand, a lot of applications directly encode data in XML. As an example, many pieces of geographic information published on the web using WebGIS applications are in the GML format. On the other hand, data in relational DBMSs, such as Oracle (Spatial), can be exported in XML.

The rest of the paper is organized as follows. In Section 2, we briefly analyze the related work. In Section 3, we describe the basic features of the ChronoGeoGraph model, with a special emphasis on its spatial and temporal components. In Section 4, we first define the XML Schema counterpart of the spatio-temporal CGG constructs; then, we give the complete translation algorithm. In Section 5, to check all constrains, we propose a spatio-temporal validator for CGG schemas. In the conclusions, we provide an assessment of the work and we outline future research directions.

2. Related Work

The study of the relationships between XML and (relational) databases is a very active research area in the XML and database communities. A lot of work has been devoted to the analysis of similarities and differences between XML and (relational) DBMSs in storing, retrieving and updating data. Moreover, various algorithms for converting data from XML to the relation model, and vice versa, have been developed. A detailed comparison of concepts available in XML schema specification languages and relational data definition languages is provided by Kappel et al. in [11]. They also define some basic correspondences between XML and relational concepts to overcome the data model heterogeneity between XML (DTDs and XML Schema) and the relational model. Finally, they propose a possible approach to the integration of XML (limited to the case of DTDs) and relational database systems. However, they do not take into consideration basic integrity constraints such as primary keys, foreign keys and cardinality constraints. Similar correspondences have been established between XML schemas and conceptual schemas. On the basis of them, various translation algorithms to map conceptual schemas into XML documents have been developed in the recent years. In particular, an algorithm to automatically generate XML DTDs from conceptual schemas, that deals with basic integrity constraints (keys and cardinality constraints), is given in [12]. The use of XML Schema for semantic data modeling is systematically investigated in [13]. In [15], Pigozzo and Quintarelli take advantage of such a characterization to devise an algorithm for generating XML Schema documents from ER schemas.

In this paper we address the problem of translating spatio-temporal conceptual schemas into XML Schema documents. The addition of spatial data to XML has been systematically studied by the Open Geospatial Consortium (OGC), which developed the standard GML. GML is an XML grammar that allows one to express geographical features. It supports a variety of geographic representations, ranging from simple geometric types, like points, lines, and polygons, to complex collections of geometries. It serves as a modeling language for geographic information systems as well as an open interchange format for geographic transactions on the internet, and it has been already adopted by a number of international projects. As for temporal information, the problem of adding one or more temporal dimensions to XML has been addressed by various researchers. As an example, the addition of a temporal dimension to XML to manage the validity of web documents has been investigated in [4], while the addition of multiple temporal dimensions to deal with normative texts in XML format is considered in [5]. In general, the hierarchical structure of XML can naturally model temporality of data. In particular, the redundancy of temporally extended relational schemas can be avoided by adopting a representation based on temporal grouping [1].

3. The ChronoGeoGraph model

The ChronoGeoGraph model (CGG for short) is a conceptual model that extends the classical Enhanced Entity-Relationship model (EER) with additional constructs for spatio-temporal information [6, 2, 3]. As for CGG features borrowed from the EER model, we refer the reader to database textbooks. We focus our attention on its spatial and temporal features.

The spatial features of CGG. First of all, the CGG model has a notion of schema territory. A *schema territory* defines the spatial domain over which all spatial elements of the schema are located. Furthermore, it supports 8 different *spatial data types*, namely, point, multipoint, line, multiline, polygon, multipolygon, collection, and undefined. Spatial data types can be associated with both entities and attributes.

As for entities, CGG distinguishes between spatial and non spatial entities. A *spatial entity* is characterized by a set of descriptive attributes plus a geometry of a given spatial data type. The geometry defines the shape and location of the spatial entity. It may be part of the primary key of the entity. A spatial entity devoid of descriptive attributes is called a *purely spatial entity*. Such entities are uniquely identified by their geometry. *Spatial attributes* are attributes that take their value over a spatial data type.

A spatial dimension can be added to relations as well. CGG distinguishes 2 different types of spatial relations: topological, metric and direction relations, and the relation of spatial aggregation. In particular, *topological relations* are spatial relations which are preserved under the topological transformations of translation, rotation, and scaling. CGG includes the set of topological relations: disjunction, adjacency, equality, inclusion, covering, overlap with disjoint borders and overlap with intersection of borders. *Spatial aggregation* expresses the composition relation over spatial entities. It constrains the geometries of the component entities to be fully included in the geometry of the compound entity; furthermore, it does not allow the geometries of the component entities to overlap and it requires the geometry of the compound entity to be completely covered by the spatial union of those of the component entities.

Besides the usual relation of specialization, CGG introduces the relation of *cartographic specialization*, which supports different spatial representations of the same spatial entity. CGG distinguishes 2 different types of cartographic specialization: (i) with scale variation, when different geometries are associated with the same entity at different scales, and (ii) with shape variation, when different geometries are associated with the same entity at a given scale. Both the parent and the children entities of a cartographic specialization are spatial entities. Besides single-level cartographic specializations based on either scale variation or shape variation, CGG encompasses two-level specialization hierarchies that pair scale variation with shape variation. It goes without saying that spatial entities may be involved in non-cartographic specializations as well. This is the case, for instance, with the specialization of the spatial entity transport network into the spatial entities road network, rail network and bus line network (which inherit the geometry of the parent entity).

Finally, CGG supports the field-based view of spatial information by the notion of (spatial) field. A *field* is a feature that varies over space. It can be associated with either the whole schema territory or a single spatial entity, and it is characterized by a specific sampling type [16].

The temporal features of CGG. CGG allows one to temporally qualify the various constructs by properly annotating them. One or more temporal dimensions can be associated with the schema territory, entities, attributes, relations (generic relations, topological relations, spatial aggregation), and fields. Different temporal dimensions are associated with different constructs. Entities can be provided with a existence time, eventually paired with a state diagram, a transaction time, an event time, and an availability time. The other constructs can be provided with a valid time, a transaction time, an event time, and an availability time.

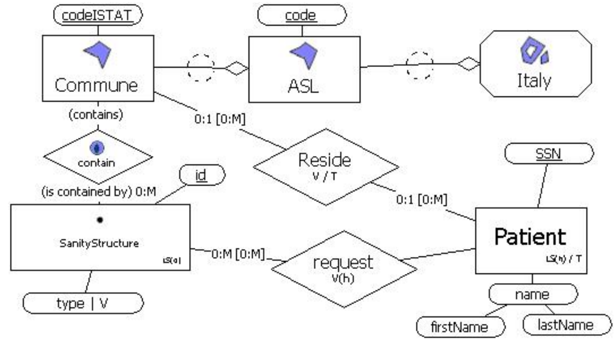

Figure 1. A simple CGG schema.

Furthermore, CGG introduces a distinction between snapshot and lifespan *cardinality constraints* for attributes and relations. *Snapshot* cardinality constraints specify the minimum and maximum number of values that an attribute can take (resp., of instances of a given entity that may participate in a relation) at *a given time*, while *lifespan* cardinality constraints specify minimum and maximum bounds with respect to the whole existence of the entity instance (resp., the validity interval of the relation instance). As for attributes, CGG also allows one to collect sets of attributes of a given entity that change in a synchronous way. Formally, a *temporal collection* is a set of entity attributes with a common temporal annotation. Finally, CGG explicitly keeps track of the events that affect a relevant element, e.g., events that change the state and/or the geometry of an entity, the validity of a relation, the value of an attribute.

Figure 1 shows the CGG schema to model a simplify Italian health-care system. This schema deals with the local medical companies (ASL) and their spatial relations with the communes , the patients and their residences, the hospitals and their localizations, and the admissions of patients to hospitals. In this application domain, spatial and temporal information play a major role. As an example, it can be used to monitor patient attraction/escape phenomena, that is, to determine the number of patients which choose to be admitted in hospitals not located in the ASL of the commune they reside in.

4. From CGG schemas to XML Schema

The encoding of basic EER constructs is straightforward. An entity E is mapped into an XML complexType element composed by the sequence of XML elements corresponding to the attributes of E. Attribute cardinalities are specified by means of the XML attributes `minOccurs` and `maxOccurs`, while the key constraint is encoded by using the XML `key` feature. Any relation R in which the entity E participates can be modeled in two different ways: either by directly including the other entities participating in R in the XML element for E or by adding to the XML element for E the references to the other entities involved in R by exploiting the `keyref` feature.

Let us consider now the encoding of CGG spatial and temporal features in XML. As a preliminary remark, it is worth pointing out that XML allows one to guarantee some, but not all, spatial and temporal constraints of a CGG schema. In particular, it cannot guarantee the constraints imposed by topological relations. As an example, it cannot constrain the geometry of a given spatial entity, e.g., a region, to be included into the geometry of another one, e.g., a country (topological relation of inclusion). In a similar way, whenever the snapshot and lifespan cardinality constraints on a given attribute differ from each other, it cannot impose both of them (the same holds for the constraints on the participation of entities in relations). The proposed translation keeps track of such constraints by properly annotating the generated XML Schema document (`annotation` feature) and delegates their verification to specific spatio-temporal validator as described in Section 5.

The spatial constructs of CGG are dealt with as follows.

Schema Territory. The schema territory defines the spatial domain of the schema and it has only one geometry instance. It is encoded in XML by adding a geometry element, called SchemaTerritory, as a subelement of the root element.

Spatial Entity. In GML, the basic components of identifiable spatial objects are defined as extensions of AbstractFeatureType obtained by adding a spatial reference. Analogously, the XML encoding of spatial entities extends that of basic entities by adding a spatial geometry reference to the sequence of attributes. In addition, the algorithm annotates the element with the indication of its topological relation with the schema territory (if it is different from the standard relation of `inclusion`).

Topological relations and spatial aggregation. CGG makes it possible to constrain, at the schema level, the topological relation that holds between the geometries of any pair of spatial entities (the same holds for spatial aggregations). Since XML Schema cannot automatically validate topological constraints on spatial relations between entities, we translate the topological relations and the relation of spatial aggregation as generic relations. In addition, we annotate the XML schema with information on the type of topological relation that holds between the participating entities. In the following example, we provide the XML counterpart of the topological relation of inclusion connecting a commune to the country (one and only one) it belongs to.

To model the relation of spatial aggregation, we add a subelement aggregation to the element for the compound entity, which includes the elements for the component entities or references to them.

Cartographic specialization. Cartographic specialization can be dealt with as the usual relation of specialization by adding as many distinct subelements as the different geometries are.

Field. Fields linked to spatial entities or to the schema territory are mapped into an XML complexType element, which includes the value and the sampling type (geometric type) associated with the field.

One or more temporal dimensions can be added to various CGG constructs. In the following, we will briefly analyze the most relevant cases. As already pointed out, in general XML Schema cannot guarantee both snapshot and lifespan cardinality constraints. We will exploit the XML attributes `minOccurs` and `maxOccurs` to respectively encode the minimum snapshot cardinality and the maximum lifespan cardinality. In addition, we will keep track of the remaining two constraints by properly annotating the XML schema.

Temporal attribute and collection. The XML counterpart of a temporal attribute (resp., a temporal collection) is an XML element that specifies the set of values that the attribute (resp., set of attributes) has taken over time. Moreover, for every such value we add a pair of XML attributes for every temporal dimension. The optional XML attribute `use` allows one not to define a CGG attribute when its value is undefined. Such a possibility is exploited to represent right endpoints of intervals (which are not defined for open intervals). To translate a temporal collection, we introduce an XML complex element which encodes the attributes of the collection, with their cardinality, as XML subelements and the temporal dimensions as XML attributes.

Temporal attribute and collection. The XML counterpart of a temporal attribute (resp., a temporal collection) is an XML element that specifies the set of values that the attribute (resp., set of attributes) has taken over time. Moreover, for every such value we add a pair of XML attributes for every temporal dimension (one or two attribute for event time). The optional XML attribute `use` allows one not to define a CGG attribute when its value can be undefined. Such a possibility is exploited to represent right endpoints of intervals (which are not defined for open intervals). To translate a temporal collection, we introduce an XML complex element which encodes the attributes of the collection, with their cardinality, as XML subelements and the temporal dimensions as XML attributes.

Temporal entity. When temporal dimensions are associated with an entity, we must add to its complex element a special subelement, called lifespan, that can be view as a temporal attribute. In general, the lifespan element is empty and its cardinality is 1:1[1:1]. If a state diagram is associated with the entity, the lifespan element of entity must be a value takes over the set of states of the state diagram and its cardinality is 1:1[1:N]. Since CGG allows one to associate the same state diagram with different entities, a `simpleType` is separately defined for every distinct state diagram, which introduces a restriction on the values that the state attribute can assume.

Temporal relation. The inclusion method cannot be applied to temporal relations. Relation changes would indeed force one to repeatedly insert all tree subelements (redundancy). All temporal relations (including aggregation) are dealt with by using `key` references. As in the case of temporal attributes, every temporal dimension is modeled as an XML attribute of the element that encodes the relation.

Temporal schema territory. Whenever a temporal dimension is added to the schema territory, we associate it with its geometry, thus reducing the problem of translating the temporal schema territory to that of translating a temporal geometry attribute.

Temporal field. In the case of temporal fields, we basically add a temporal dimension to the sampling. Temporal fields are then dealt with by introducing temporality in XML field elements, analogously to the case of temporal compound attributes.

Event. CGG explicitly models events that initiate/terminate valid time intervals as well as events that initiate/terminate availability time intervals. In XML, events are modeled as additional temporal attributes. We introduce the attributes `startEventVT` and/or `endEventVT` for the events that affect valid time intervals, and the attributes `startEventAT` and `endEventAT` for the events that affect availability time intervals. Moreover, we constrain these attributes to take their value over the set of admissible event types specified by the CGG schema.

Now, we conclude the section by sketching the complete algorithm that maps a CGG schema into the corresponding XML Schema document. It basically pairs the rules for basic ER constructs, defined by Pigozzo and Quintarelli in [15], with the above-described additional rules needed to deal with the spatio-temporal features of CGG schemas. The algorithm takes advantage of the following notions of first-level entity and inclusion condition, which extend the corresponding notions introduced in [15]. A *first-level entity* (FLE) is an entity that satisfies one of the following conditions: (i) its (snapshot) participation in relations is always partial; (ii) it totally participates only in many-to-many relations and in one-to-many relations (on side one); (iii) it is the parent entity of a specialization and it is not involved in any other relation; (iv) it plays the role of compound entity in an aggregation and it is not involved in any other relation. FLEs are coded as elements of the root. A one-to-one or one-to-many relation R can be represented as an element of the participating entity E_1 (on side one, in the case of one-to-many relations) that specifies the other participating entity E_2 provided that the following conditions, called *inclusion conditions* (IC), hold: (i) E_2 has not been coded yet; (ii) E_2 totally participates in R; (iii) if R is not the identifying relation, E_2 is not weak entity; (iv) Ea_2 is not a child of a specialization; (v) E_2 is not a component entity of an aggregation; (vi) E_2 is not a first-level entity; (vii) R is not temporal relation. If some of these conditions are not satisfied, we only add a reference to entity E_2 in entity E_1 using the `keyref` feature. An analogously condition is defined for aggregations.

The main steps of the translation algorithm can be summarized as follows:

- create the file .xsd and define default namespaces in the xs:schema element which includes the gml import and the root element of XML counterpart of the CGG schema (if the CGG schema has not a spatial component, gml is not imported);
- create the root element for the CGG schema and, if it has spatial components, include the schema territory;
- identify the first-level entities (FLEs);
- for every entity (first the first-level ones and then the other ones), execute the following steps (entity management procedure):
    a. if it has not been already encoded, encode it as a subelement of the current element;
    b. for any relation it participates in which has not been already encoded (if any), execute one of the following actions (the choice of the action to execute depends on the structural properties of the relation [15]): (i) encode the relation as a subelement of the considered entity, and either include the other entities participating in the relation as subelements of such a relation element or simply add a reference to them (in the case of one-to-one or one-to-many relations, the choice depends on the truth or falsity of the inclusion condition; in the case of many-to-many relations, the second option is always selected); (ii) do nothing and delegate the encoding of the relation to one of the other participating entities;

c. for any aggregation it participates as compound entity, encode it as a subelement of the considered entity, and either include the component entities as subelements of such an element or simply add a reference to them (the choice depends on the inclusion condition for aggregation);

d. for any specialization it participates as the parent entity, encode it as a subelement of the considered entity, and either include the child entities as subelements of such an element;

e. keep track of the entity identifier (key);

f. encode the temporal collections (if any);

g. encode the descriptive attributes which do not belong to any temporal collection (if any);

h. if the entity is spatial, encode its geometry (provided that it does not belong to a temporal collection) and fields (if any);

i. if the entity is temporal, encode its temporal dimensions;

j. for each entity subelement included from b to d recursively call the translation procedure;

- for every field of CGG schema include relative element;
- include the keys and `keyrefs`, which have been determined by the entity management procedure, in the root element for the CGG schema;
- include the `simpleTypes` associated with the state diagrams specified by the CGG schema (if any) in the xs:schema element.

From the simple CGG schema described in Figure 1, the proposed algorithm, as a first step, creates the XML Schema file and it specifies its initial elements. Since the CGG schema is spatial, it adds a geometry subelement, that represents the schema territory, to the root element (the `Sanity` element in our example). Then it determines the first-level entities, namely, `Patient` and `ASL`. Next, it starts the entity management procedure. The first entity to be dealt with is the entity `Patient`. The algorithm creates an `Patient` element as a complexType. It participates in two relation... The next first-level entity is `ASK`, and the algorithm repeats on it the same operations it applied before. When all entities in the CGG schema are encoded in the XML Schema document, `keys` and `keyRefs` are included in the Sanity element. Finally, the `simpleType` associated with state diagrams are specified. The result of the execution of the algorithm is given by the hierarchical structure of Figure 2.

5. Spatio-Temporal Validation

This section describes the spatio-temporal validation library. The library is used to validate XML documents (also referred to as XML instances) with respect to XML schemas encoding CGG conceptual schemas. The constrains that one may want to check are of three types:

1. *Constrains on the CGG model*. These are constrains that refer to the CGG model. For instance, a constrain that forces a topological relation to relate spatial entities only. These constrains are checked when the CGG model is synthesized and hence are not part of the validation process;
2. *W3C XML Schema Language constrains*. These are constrains that can be encoded directly in the XML schema language. For instance, to bound the minimum and maximum number of occurrences of a multivalue attribute we can use the `minOccurs` and `maxOccurs` attributes of the `xs:element` element. These constrains are verified by using a standard validator for the W3C XML Schema Language.
3. *Spatio-temporal constains*. These are all spatial and temporal constrains defined in the CGG conceptual model that are not checked in the previous two steps. For instance, the fact that a certain geometric entity is contained in another geometric entity or that any validity interval is a proper temporal interval. These constrains are checked by the developed validation library.

The general architecture of the spatio-temporal validator is depicted in Figure **??**. The validator is divided in three main packages:

1. *JDOM package*. It contains classes to parse the XML instance and the XML schema into DOM models. It uses the Java parser JDOM [9].
2. *Xerces package*. It contains classes to check the W3C XML Schema Language constrains as defined in [17, 18]. It uses the popular XML schema validator Xerces [20].
3. *Spatio-temporal validation package*. It contains classes to fetch the spatio-temporal constrains from the XML schema document, to retrieve the corresponding constrain instances from the XML instance, and to perform the actual validation. The retrieval of the constrain (instances) is performed by taking advantage of the XPath processing library Jaxen [8]. The validation of spatial constrains involving knowledge of computational geometry notions is performed with the aid of the library Java Topological Suite [10].

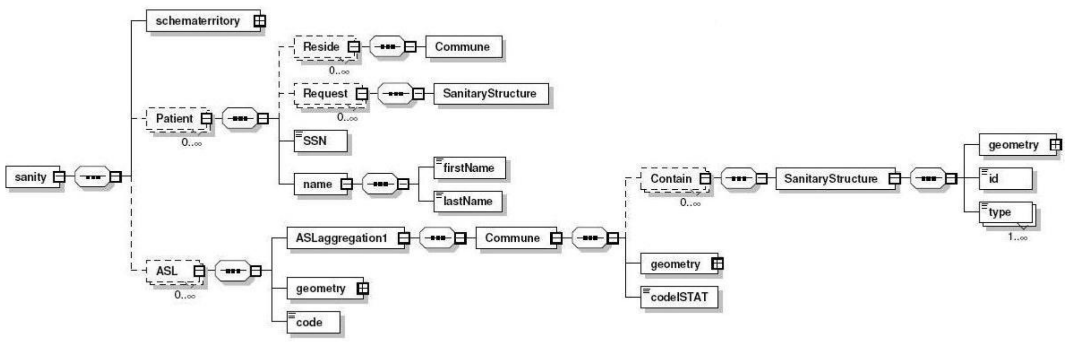

Figure 2. The XML schema corresponding to CGG schema of Figure 1.

The Java source code for the validation library and the related Javadoc documentation is available at the CGG project website [**?**].

6. Conclusions and further work

In this paper we outlined a translation algorithm that maps spatio-temporal conceptual schemas, expressed in the CGG formalism, into XML schema documents. It directly encodes in XML Schema all basic integrity constraints, e.g., domain, primary key, foreign key, and snapshot cardinality constraints; moreover, it keeps track of additional spatial and temporal constraints by taking advantage of XML attributes and annotations. The algorithm has been implemented and integrated in a software tool for the synthesis of CGG schemas [7].

References

[1] J. Clifford, A. Croker, F. Grandi, and A. Tuzhilin. On temporal grouping. In *Proceedings of the International Workshop on Temporal Databases*, pages 194–213, London, UK, 1995. Springer-Verlag.

[2] C. Combi and A. Montanari. Data models with multiple temporal dimensions: Completing the picture. In Proc. of the 13th International Conference on Advanced Information Systems Engineering (CAiSE), pages 187–202, London, UK, 2001. Springer.

[3] I. De Fent, D. Gubiani, and A. Montanari. Granular geograph: a multi-granular conceptual model for spatial data. In Proc. of the 13th Italian Symposium on Advanced Database System (SEBD), pages 368–379, Bressanone, Bolzano, Italy, 2005.

[4] F. Grandi and F. Mandreoli. The valid web: An XML/XSL infrastructure for temporal management of web documents. In Proc. of the 1st International Conference on Advances in Information Systems (ADVIS), pages 294–303, Izmir, Turkey, 2000.

[5] F. Grandi, F. Mandreoli, P. Tiberio, and M. Bergonzini. A temporal data model and management system for normative texts in XML format. In Proc. of the 5th ACM international workshop on Web information and data management (WIDM), pages 29–36, New York, NY, USA, 2003. ACM Press.

[6] D. Gubiani and A. Montanari. ChronoGeoGraph: an expressive spatio-temporal conceptual model. Submitted for publication. 2007.

[7] D. Gubiani and A. Montanari. A tool for the visual synthesis and the logical translation of spatio-temporal conceptual schemas. Submitted for publication. 2007.

[8] Jaxen. Available from `http://jaxen.org/`. As viewed on March 28, 2007.

[9] JDOM. Available from `http://www.jdom.org/`. As viewed on March 28, 2007.

[10] Java Topology Suite. Available from `http://www.vividsolutions.com/jts/`. As viewed on March 28, 2007.

[11] G. Kappel, E. Kapsammer, S. Rausch-Schott, and W. Retschitzegger. X-Ray - towards integrating XML and relational database systems. In Proc. of the 19th International Conference on Conceptual Modeling (ER), pages 339–353, 2000.

[12] C. Kleiner and U. W. Lipeck. Automatic generation of xml dtds from conceptual database schemas. In *GI Jahrestagung (1)*, pages 396–405, 2001.

[13] M. Mani, D. Lee, and R. R. Muntz. Semantic data modeling using XML schemas. In Proc. of the 20th International Conference on Conceptual Modeling (ER), pages 149–163, Yokohama, Japan, 2001.

[14] OGC. *OpenGIS Geography Markup Language (GML) Encoding Specification*, 2006.

[15] P. Pigozzo and E. Quintarelli. An algorithm for generating xml schemas from er schemas. In Proc. of the 13th Italian Symposium on Advanced Database Systems (SEBD), pages 192–199, Bressanone, Bolzano, Italy, 2005.

[16] P. Rigaux, M. Scholl, and A. Voisard. *Spatial Databases with an Application to GIS*. Morgan Kaufmann Pub., 2002.

[17] World Wide Web Consortium. XML Schema Part 1: Structures. `http://www.w3.org/TR/xmlschema-1/`, 2004.

[18] World Wide Web Consortium. XML Schema Part 2: Datatypes. `http://www.w3.org/TR/xmlschema-2/`, 2004.

[19] World Wide Web Consortium. XQuery 1.0: An XML Query Language. `http://www.w3.org/TR/xquery`, 2007.

[20] Xerces. Available from http://xerces.apache.org/xerces-j/. As viewed on March 28, 2007.