A Tool for Transforming Conceptual Schemas of Spatio-Temporal Databases with Multiple Representation

Mohammed Minout
Department of Informatics & Networks
Université Libre de Bruxelles
1050, Brussels, Belgium
mminout@ulb.ac.be

Christine Parent
INFORGE
Université de Lausanne
CH 1015 Lausanne, Switzerland
christine@lbd.epfl

Esteban Zimányi
Department of Informatics & Networks
Université Libre de Bruxelles
1050, Brussels, Belgium
ezimanyi@ulb.ac.be

Abstract

Nowadays, classical conceptual models (such as ER or UML) are used for designing database applications. These classical conceptual models usually come with associated CASE tools assisting the user from the creation of the conceptual schema until the generation of a physical schema for a relational or an object-relational DBMS. However, when developing spatial or temporal databases such classical models are inadequate since they do not consider spatio-temporal concepts. Although spatial and/or temporal extensions have been proposed for these models, such extensions do not cope with the requirements of advanced geographical applications, in particular, they do not cope with multiple representations of the same real-world phenomenon. In the context of the European project MurMur we developed a conceptual model called MADS coping with spatio-temporal information having multiple representations. Typical examples of multiple representation arise with multi-scale or time-varying information. The MADS model is supported by a set of associated tools allowing both the definition of the schema and the queries of the application at a conceptual level. Such conceptual specifications are then automatically translated into the language supported by the target GIS and/or DBMS software (e.g., SQL schema definitions or queries for an Oracle database). In this paper we describe a Translator module that allows to implement conceptual MADS schemas into target platforms. It is composed of a Transformation module that translates conceptual MADS schemas into a more simple schema using only concepts supported by the target software, and a Wrapper module that generates the data structures in the language of the GIS and/or DBMS software.

Keywords: Conceptual Modeling, Spatio-Temporal Databases, Logic Design, Geographic Information Systems.

1 Introduction

Applications manipulating geodata are difficult to model due to the particularity and complexity of the spatial and temporal components. More facets of real-world entities have to be considered (location, form, size, time validity), more links are relevant (spatial and temporal links), several spatial abstraction levels often need to be represented. Thus, modeling spatio-temporal databases requires advanced facilities, such as the following:

- Objects with complex structure (e.g., non-first-normal form), generalization links, composition/aggregation links, at least equivalent to those supported by current object-oriented models. This should achieve full representational power in terms of data structures.
- Spatial objects with one or several geometries associated to different resolutions or user viewpoints;
- Alternative geometry features to support both discrete and continuous views of space.
- Temporal objects with complex lifecycle that allow users to create, suspend, reactivate, and eventually delete objects.
- Timestamped attributes that record their past, present, and future values.
- Spatio-temporal concepts for describing moving and deforming objects.
- Explicit relationships to describe structural links as well as spatial (such as adjacency, inclusion, spatial aggregation, ...), and synchronization links (such as before, during, ...). The knowledge of the topological links between real-world entities is an essential requirement for applications.
- Causal relationships describing the causes and effects of changes that happen in the real world.

The model must also allow defining schemas that are readable and easy to understand. A key element for achieving this double objective is the orthogonality of the structural, temporal, and spatial dimensions of the model (and more generally of the concepts of the model). Thus, whatever the concept of the model (e.g., object, relationship, attribute, aggregation), the spatial and temporal dimensions may be associated to it.

This work focuses on logical and physical design for spatio-temporal databases. This is one step of a complete design methodology for applications coping with spatial and/or temporal data having multiple representations. The contribution of this work is two-fold. (1) The presentation of MADS, a conceptual spatio-temporal model with multiple representations. The MADS model includes the advanced modeling facilities described previously. Such facilities have been derived from a generic analysis of the requirements of typical spatio-temporal applications. (b) The translation of these concepts into a logical model, called MADSLog, which is platform independent, and then into a physical model targeted to specific implementation platforms (DBMSs or GISs).

The remainder of the paper is organized as follows. Section 2 presents the MADS data model. In Section 3, a small set of transformation rules for the multi-representation, spatial, and temporal dimensions is given. Section 4 presents the architecture and design of the Translator tool. Section 5 concludes and points to directions for future research.

2 The MADS Data Model

MADS is a conceptual spatio-temporal model based on an extended ER model. MADS stands for Modeling of Application Data with Spatio-temporal features. In [2] the authors analyze different spatio-temporal data models along the axes of expressiveness, simplicity and comprehensiveness, formalism, and user friendless, making the conclusion that none of the existing models satisfied all the criteria.

MADS includes a set of predefined spatial and temporal Abstract Data Types (ADTs) that are used for describing the spatial and temporal characteristics of schemas. Similarly, MADS provides a set of ADTs supporting multi-representation features. In MADS the structural, spatial, temporal, and multi-representation modelling dimensions are orthogonal, meaning that spatial, temporal and multi-representation features can be freely added to any construct of the schema (object and relationship types, aggregation links, attributes, . . .).

2.1 Spatial and Temporal Dimensions

The concept of spatiality covers the notions of shape and location. Shape describes the geometric form associated with the representation, generally a point, a line or an area. The location allows to situate this form in space. To describe the spatiality of real-world phenomena represented in the database, MADS provides several Spatial Abstract Data Types (SADTs) [4] organized in a generalization hierarchy [3]: generic (Geo), simple (Point, Line, Simple Area, Oriented Line), and complex types (Point Set, Line Set, Complex Area, Oriented Line Set). The generic type Geo means only “this type is spatial”. The definition of the precise type of each occurrence will be done at the time of its creation. Each spatial type has an associated set of methods to define and handle the instances of this type.

MADS also enables to describe continuous fields with the concept of space-varying attribute, i.e., attributes whose values are defined by a function having as domain the set of points of any non-punctual spatial extent, i.e., any SADT value but a unique point. Spatial phenomena described by space-varying attributes can be continuous (like elevation or temperature), stepwise (like the type of crop in a cultivated area), or discrete (like mines in a minefield). Each type of function defines the kind of interpolation, if any, that can be done to compute the value(s) of the attribute.

Temporality associated to object or relationship types concerns the existence of instances in their type. Thus, temporality describes their lifecycle: objects are created, can be temporarily suspended, then reactivated, and finally removed. The lifecycle is described by a particular time-varying attribute, Status, which can take one of the four values: not-yet-existing, active, suspended, or disabled. An attribute is said to be time-varying if its values change over time, while keeping track of the changes. Like space-varying attributes, time-varying attributes are defined as a function of the time to the value domain. Keeping track of rainfall values in a given area over time is an example of information varying simultaneously in both space and time. The different values of the attribute over time are conserved, and each value is associated with a temporal element that describes its validity as seen by the application (valid time) or known to the database (transaction time). This temporal element is described by using one of the Temporal Abstract Data Types (TADTs) in MADS, which are also organized in a hierarchy.

2.2 Multiple Representations

There is no natural unique way to look at some phenomenon and there is no natural unique way to represent it. Usually, the same database must serve many different purposes, each of them requiring different data or the same data but with different representations. The multiple representation facilities supported by the MADS data model, rely on a simple idea [8]: the possibility to stamp any element of a schema with a representation stamp (or r-stamp) that identifies for which scale, time frame, viewpoint, etc. the element is valid. Object types, relationship types, methods, attributes (including geometry and lifecycle) can be r-stamped and therefore have different representations.

Representations may vary according to different criteria. In the MurMur project, we focused on two criteria. The first, viewpoint, is the materialization of user’s needs. For example different user profiles see different information in the database. The second, resolution that specifies the level of detail of a representation. However, any other criteria can be used for defining the representations. Stamping may apply on data, whether it is object and relationship instances, or attribute values, and on meta-data, whether object and relationship type definitions or attribute definitions. Stamping allows users to personalize object and relationship types by changing their attribute composition according to their stamps, and to keep several values for the same attribute.

2.3 Constrained Relationships

MADS constrained relationship types are relationship types that convey spatial and temporal constraints on the objects they link. MADS includes topological and synchronization relationships as built-in constrained relationship types. For example, a topological relationship type Inside may be defined to link object types Station and River, expressing that the geometry of a Station is within the geometry of the related River. There are at least three constrained types for spatial relationship: topological, orientation, and metric. Each type defines a spatial constraint between the geometry of the objects linked. The MADS model proposes a range of predefined topological relationships, such as disjunction, adjacency, intersection, cross, inside, and equal.

Synchronization relationships allow specifying constraints on the lifecycles of the participating objects. They convey useful information even if the related objects are not timestamped. They allow in particular to express constraints on schedules of processes. For the semantics of the definition of synchronization relationships, MADS uses the predefined operators proposed by Allen [1], namely: before, equals, meets, overlaps, during, starts, and finishes.

We illustrate the concepts explained in the previous paragraphs with the example shown in Figure 1. It shows

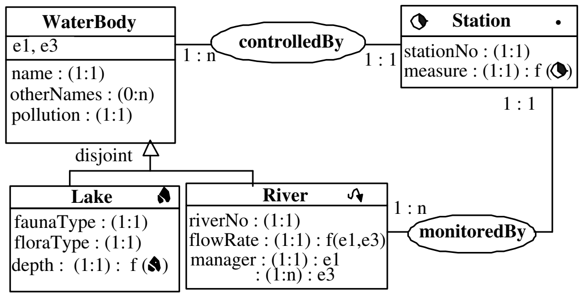


Figure 1. An example of a MADS schema.

a MADS schema for a river monitoring application. In MADS schemas, the spatial, temporal, and representation characteristics are visually represented by icons. We refer to [8] for the complete list of the spatial and temporal types available in MADS.

Temporal and spatial icons are embedded into object or relationship types. The temporal icon (symbolizing a clock) on the left-hand side of the object/relationship type expresses that the lifecycle information is to be kept. Spatial icons are shown on the right-hand side. The example schema shows the object types River and Lake that are subtypes of WaterBody. The Station type keeps water quality information about the rivers. The stations may not operate continuously: their lifecycle information allows to records periods of operation. Each station provides several measures, whose validity period is recorded. Each lake records space-varying information of depth. The WaterBody object type has two representations e1 and e3. The manager attribute has two definitions, one for e1 and another for e3. Both definitions of manager have the same domain but have different cardinalities. One value is kept for the stamp e1 and several values for stamp e3.

3 Transformation Rules

MADS is a spatio-temporal data model explicitly obeying the orthogonality principle in adding space, time, and multiple representation features to data structures. Therefore, the transformations of an element of the schema (for example a spatio-temporal object type) according to each modeling dimension are independent of the characteristics of the element on other dimensions. Thus, the transformation rules could be defined for each dimension, independently of the others.

The following paragraphs present some translation rules for each modeling dimension. As the rules for translating the structural dimension are well known (e.g., [5]), we do not discuss them in this paper.

3.1 Rules for Multiple Representation Transformations

In a database with multiple representations, each element of the schema (object type, relationship type, attribute, and method) has an r-stamp defining its visibility. This r-stamp is defined either explicitly by the designer of the database, or implicitly as being the same as that of the element to which it belongs (e.g., an object type belongs to its schema, a component attribute belongs to its composite attribute).

There is a set of rules for transforming multiple representations. We restrict in this paper to one example of transformation, the rest of these rules are given in [3].

The purpose of this rule is to transform an attribute or method having several definitions (each one associated with an r-stamp) into several attributes (one attribute by definition). Indeed, target systems (e.g., Oracle 9i, Arc View, etc.) do not allow neither a type having two attributes or methods of the same name, nor an attribute or a method having two definitions. Figure 2 shows the application of this transformation on the manager attribute that has two definitions in the initial schema: monovalued for the stamp e1, multivalued for the stamp e3.

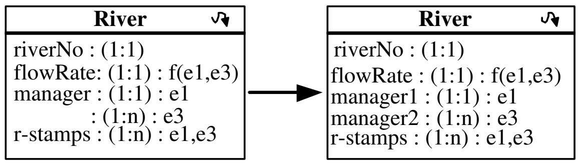


Figure 2. Transformation of manager attribute.

3.2 Rules for Spatial Transformations

MADS enables the designer to attach spatiality both to object or relationship types and to attributes. As already said, MADS provides a set of Spatial Abstract Data Types, organized in a generalization hierarchy.

3.2.1 Spatial Types

An spatial object or relationship type represents phenomena whose spatial reference is relevant for the application. For the target systems that do not have the concept of spatial type, but have spatial attributes, such as Oracle 9i, a spatial type is transformed by creating a monovalued spatial attribute, called geometry, having the same spatiality as the initial object type (see Figure 3).

3.2.2 Space-Varying Attributes

Although many geographical applications use space mainly in a discrete way, there is often a need to describe continuous fields, such as elevation or land use. A continuous field may be perceived in two different ways. The usual view is that it describes the values of a variable over the whole space. This is the approach chosen by [6, 7] and by all GISs offering a continuous view of the space (raster GIS). Another view is that a continuous field describes the values of an attribute over the geometry of an object type. This is the approach chosen in MADS, both for spatial objects and for spatial attributes with non-zero dimension. Space-varying attributes are represented using the function icon f( ). There are three types of functions as mentioned in Section 2.1. In this rule, the function used is continuous.

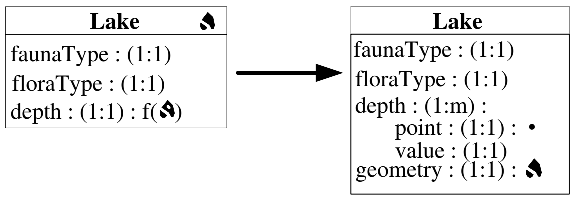


Figure 3. Transformation of spatial object type Lake and the space-varying attribute depth.

For systems that do not support the concept of space-varying attribute, the transformation consists in replacing the attribute by a multivalued complex attribute composed of a spatial element and a value. Figure 3 shows this transformation for the space-varying attribute depth, composed of a point and the value of the attribute at that point.

3.3 Rules for Temporal Transformations

MADS allows to associate temporality to any concept of the model (object type, relationship type, and attribute). Temporality expresses the lifecycle of object or relationship types, and the validity of values for time-varying attributes.

3.3.1 Lifecycle of Temporal Types

In MADS, a temporal object (or relationship) type keeps track of the lifecycle of its instances, defined by the events of creation, suspension, reactivation, and destruction. For each instance, this information remains available after its destruction as its value (current value for nontemporal attributes, history of values for temporal attributes), so it can be able to associate information on the lifecycle of the object type corresponding of the real world.

In a similar way to the transformations for spatiality, the lifecycle of temporal types (Figure 4), is transformed by creating a monovalued time-varying attribute, called status, which can take as a value one of the following four states: not-yet-existing, active, suspended, or disabled.

This rule also generates a set of temporal integrity constraints associated with the states. For example, an instance cannot be active and suspended at the same time. Another integrity constraint specifies the temporal type associated with the validity of the active value: an instant for a temporal type describing an instantaneous event (and identified as such by the instant icon for its lifecycle), an interval for a temporal type having a coninuous lifecycle, a set of intervals for a temporal type that can be suspended.

3.3.2 Time-Varying Attributes

This transformation contains two rules. The first rule works

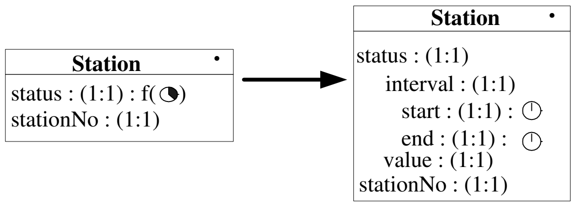


Figure 4. Transformation of attribute interval.

as the transformation of a space-varying attribute in changing the variable attribute depth by status and the point by interval. This rule generates an integrity constraint expressing that the intervals must be disjoint and the values must belong to the domain of the status, namely one of the states not-yet-existing, active, suspended, or disabled. However, most of existing systems have a domain of the instant type (for example the DATE type), but not a domain of the interval type of time. The second rules thus transforms an interval into a complex attribute having the same name, the same cardinality, and whose component attributes (start and end) are of instant type, as shown in Figure 4. This rule generates a integrity constraint expressing that for each value of the attribute interval, interval.start must be less than or equal to interval.end.

4 Translator Tool

4.1 Design and Implementation

The Translator tool allows to translate conceptual MADS schemas into physical schemas that can be implemented on operational systems. Since the target implementation platforms have different expression power (e.g., relational DBMSs do not support multivalued attributes while object-relational DBMSs do), this translation process uses an intermediate logical model, called MADSLog, containing all concepts of MADS and a few logical concepts such as the reference attribute. The use of the MADSLog model allows making the translator extensible, minimizing the effort for adding new target platforms.

For each target GIS and DBMS, a subset of the MADSLog data model is defined containing all the concepts of MADSLog that have an equivalent in the target data model. For instance, in Oracle 9i using an object-relational model, multivalued attributes may be represented using the NESTED TABLE and VARRAY concepts. However, in Oracle 9i using a classical relational model a multivalued attribute has be transformed into a new object type and a new relationship type. Therefore, for each target GIS and DBMS it must be determined which rules will be applied while generating the corresponding MADSLog schemas.

The Translator tool is composed of two modules. First, a Transformation module that transforms a MADS schema into a MADSLog schema adapted to the target GIS and DBMS (e.g., Oracle 9i). Second, a set of specialized modules, called Wrappers, one for each target GIS and DBMS. A Wrapper rewrites the MADSLog schema provided by the Transformator, and generates a physical schema expressed in the language of the target system (e.g., SQL scripts for Oracle 9i).

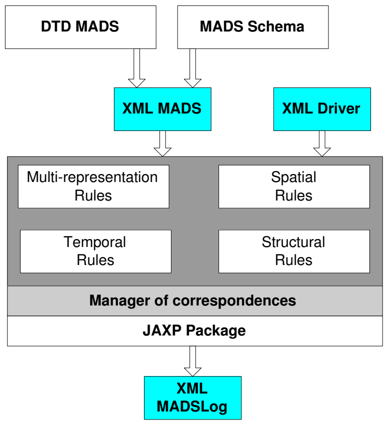
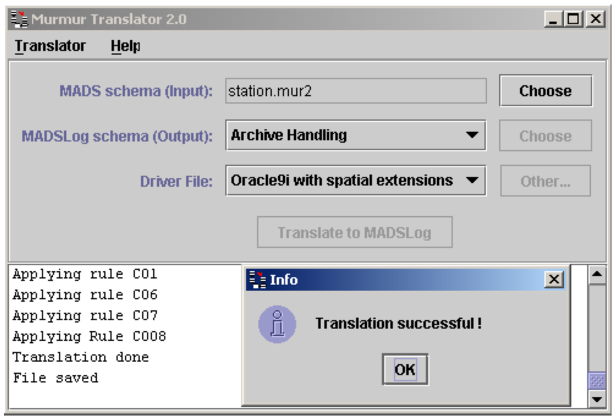


Figure 5. Architecture of the Transformation module.

The architecture of the Transformation module is shown in Figure 5. The core of the Transformation module is composed of all transformation rules for the structural, spatial, temporal, and multiple representation dimensions. Associated to each target GIS and DBMS, there is a Driver defining which rules will be applied while generating MADSLog schemas. Finally, another module keeps the correspondence between each element of the original MADS schema and its translation in the MADSLog schema. Such correspondences are used for translating queries.

We use XML as an exchange format for expressing schemas. Each MADS schema produced by a graphical schema editor is exported in XML format respecting a Data Type Description (DTD). The different drivers selecting the set of transformation rules to be activated are also expressed in XML.

The Translator has been implemented using the Java platform. A screen capture is shown on Figure 6. We used the JAXP (Java API for XML Parsing) package to implement the transformation rules. The JAXP package enables applications to parse and transform XML documents using an API that is independent of a particular XML processor implementation.



MADS schema (Input): station.mur2
MADSLog schema (Output): Archive Handling
Driver File: Oracle9i with spatial extensions

```
Applying rule C01
Applying rule C06
Applying rule C07
Applying Rule C008
Translation done
File saved
```

Translation successful !

Figure 6. Interface of the MurMur Translator

The MADSLog schema is produced by applying successively the transformation rules to each element of the schema. As shown in Figure 4, the same element can be successively transformed by several rules. The total order for applying the rules is as follows: first, rules for multiple representation, then those relating to spatial and temporal dimensions, and finally the structural rules.

The tool works recursively: it applies to each node of the schema all the rules defined in the driver. If a rule is activated (is applied to the node), the process restarts again from the root of the schema. The process stops when no transformation rule is applicable to the target schema.

4.2 Example of Use

We take an excerpt of the example in Figure 1 (the object type Station). The transformation of this conceptual MADS schema into a MADSLog schema targetting an object-relational model is given in Figure 7. The MADSLog schema defines the Station object type with attributes station, statusNo, and geometry, as well as the DStatus domain, representing the lifecycle.

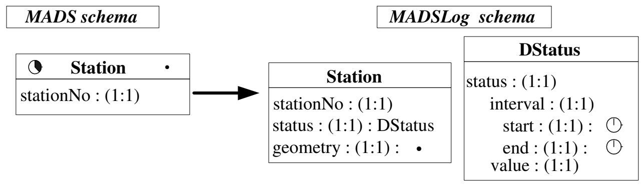


Figure 7. MADS and MADSLog schemas.

Finally, the Wrapper translates this MADSLog schema into an SQL script for Oracle 9i as follows:
CREATE OR REPLACE TYPE DStatus_Int AS OBJECT (start DATE, end DATE);
CREATE OR REPLACE TYPE DStatus_Value AS OBJECT (interval DStatus_Int, value FLOAT)
CREATE OR REPLACE TYPE Status_Seq AS OBJECT (status DStatus_Value, sequence INTEGER);
CREATE OR REPLACE TYPE DStatus AS TABLE OF Status_Seq;
CREATE OR REPLACE TYPE DId AS OBJECT (value INTEGER);
CREATE OR REPLACE TYPE DStation AS OBJECT (id DId, stationNo VARCHAR2(12), status DStatus, geometry MDSYS.SDO_GEOMETRY);
CREATE TABLE Station OF DStation NESTED TABLE status STORE AS StationStatus;

5 Conclusion and Future Work

Schema translation is important for providing integration and interoperability in multiple database systems. Two shortcomings of many of the existing approaches to the problem are that they are not easily automated and that they lack a formal basis for evaluating the correctness of the resulting translations. We have discussed an approach based on structural, spatial, temporal, and multiple representations transformations that overcomes both of these problems.

This paper presents a tool for translating conceptual schemas of spatio-temporal databases with multiple representation. The ultimate goal of the tool is to be a universal translator from the MADS conceptual model into any operational data model of GIS or DBMS. The translation is made in two steps. First, translation of a MADS schema into a less expressive MADS schema, called MADSLog, involving only the concepts available at the target data model. This translation generates a set of spatio-temporal integrity constraints. Second, translation of this MADSLog schema into a schema expressed with the syntax of the target system. This two-steps strategy allows to reuse most of the code when adding a new target data model.

We continue to study the problem of schema translation, in particular the translation of integrity constraints, implemented using triggers in target DBMSs. Further, we are also implementing a tool allowing automatic translation of visual queries expressed over the conceptual MADS schema into queries of the target DBMS or GIS (e.g., SQL queries for Oracle).

References

[1] J.F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, November 1983.

[2] C. Parent, S. Spaccapietra, and E. Zimányi. Spatio-temporal conceptual models: Data structures + space + time. In *Proc. of the 7th ACM Symposium on Advances in Geographic Information Systems, GIS'99*, pages 26–33, Kansas City, USA, 1999.

[3] C. Parent, E. Zimányi, M. Minout, and A. Aissaoui. Implantation d'un modèle conceptuel avec multi-représentation. In A. Ruas, editor, *Traité IGAT sur la représentation multiple et la généralisation*, chapter 7, pages 131–147. Hermès, 2002.

[4] P. Rigaux, M. Scholl, and A. Voisard. *Introduction to Spatial Databases: Applications to GIS*. Morgan Kaufmann, 2000.

[5] H. Tardieu, A. Rochfeld, and R. Colletti. *La méthode Merise: Principes et outils*. Editions d'Organisation, 2000.

[6] N. Tryfona and T. Hadzilacos. Logical data modeling for spatio-temporal applications: Definitions and a model. In *Proc. of the Int. Database Engineering and Applications Symposium, IDEAS'98*, pages 14–23, Cardiff, U.K., 1998. IEEE Computer Society.

[7] N. Tryfona, D. Pfoser, and T. Hadzilacos. Modeling behavior of geographic objects: An experience with the object modeling technique. In *Proc. of the 8th Int. Conf. on Advanced Information Systems Engineering, CAiSE'97*, pages 347–359, Barcelona, Spain, 1997. LNCS 1250, Springer-Verlag.

[8] C. Vangenot. Supporting decision-making with alternative data representations. *Journal of Geographical Information and Decision Analysis*, 5(2):66–82, 2001.