Modeling geographic data with multiple representations

Sandrine Balley
Laboratoire COGIT
Institut Géographique National
sandrine.balley@ign.fr

Christine Parent
Inforge
University of Lausanne
christine.parent@unil.ch

Stefano Spaccapietra
Database Laboratory
EPFL
stefano.spaccapietra@epfl.ch

Abstract

The impact of resolution has been widely studied on raster data, exploring new ways to store and use raster data at different resolution levels. A similar amount of work has not yet been achieved for vector data. The aim of this paper is to propose a framework suited for vector data, enabling their use at different resolution levels. The framework has in fact a wider scope as it has been designed for the general problem of using multiple representations of geographical and thematic data.
This problem can be divided into three main issues: data modeling, data matching from different data sets, and data utilization. How can geographical phenomena with several representations be modeled in vector geographic information systems (GIS)? How should different representations be matched to make the data management systems aware that they all represent the same geographical phenomenon? How can systems supporting multiple representations be used? Answers to these questions have been elaborated as part of the European MurMur project. They are presented here and illustrated using current applications from the French national mapping agency (IGN).

1. Introduction

Awareness that geographical data has become an essential source of information for an ever increasing number of applications, in particular in the public domain, has generated a number of national and international initiatives to set up global infrastructures for the collection and dissemination of geographical data: e.g., (ANZLIC 1996) (FGDC 1997) (GeoConnections 2001) (INSPIRE). Data collection efforts, however, are hampered by the difficulty to reconcile and integrate information from the many existing heterogeneous data sets. Heterogeneity has its source in both semantic aspects (because of the diversity of application requirements addressed by existing data sets) and technical aspects (because of the diversity of data management software in use). Semantic interoperability issues (Parent and Spaccapietra, 2000) are addressed using semantic mediation techniques (Wiederhold 1992), nowadays mostly supported by reference domain-specific ontologies (Guarino 1998). Technical issues are addressed by developing interoperability standards (e.g., GML (OpenGIS) and mapping tools (e.g., (FME)). The ultimate goal of all these undertakings is to enable the consolidation of consistent information about a given region (e.g., a county) from complementary or overlapping data available in several source databases. This provides a global vision of the region that encompasses geographical, social, economical, and communication data, to name but a few aspects.

Each of the original data sets maintains a specific representation of the real world, tailored by the perception and interests of the associated applications. Building a global vision inevitably results in the coexistence, within this vision, of several heterogeneous representations of the real world. A further challenge is then assessing and maintaining the consistency of the vision and of the associated data. Geographical databases add to this heterogeneity of representation as an additional factor, the diversity in the resolution of the geometric features. As different applications need data at different resolution levels, an integrated data set is likely to contain so-called multi-scale data (Steel and Worboys 1998) (Timpf 1998).

Unfortunately, current data management systems (DBMS or GIS) do not provide adequate functionality to manage multiple representations of geographical phenomena.

Relational systems offer powerful view mechanisms to accommodate diversity of perception, but as views are kept separate from the underlying schema and from each other the global vision is somewhat fragmented (the schema does not indicate which item is used by which view) and navigating among views is not easy. Object-oriented systems provide richer descriptive semantics, but they only support simple filtering views (no projection, no restructuring) because of the inherent contradiction between their inheritance and sub-typing rules (Ye et al. 1997). GIS do not go much beyond sometimes supporting multiple geometries for objects. Research proposals have addressed some facets of multiple representation support (e.g., views, roles, time-stamping, multi-scale data structures), but do not attempt to provide a generic approach that might also apply to the other facets of the problem.

In summary, it is still an open question how to develop systems that can properly and generically handle multiple representation and multiple resolution databases, and how to handle the corresponding consistency issues, thus ensuring substantial savings in updating costs thanks to update propagation techniques (Badard 1999). The work reported in this paper is a contribution in this direction. Basically, the paper introduces new data modeling concepts that seek to make the system aware of the existence of multiple representations and thus overcome current limitations. Once the system knows about multiple representation, it may enforce consistency checking and update propagation based on explicitly defined rules that are stored as part of the data description. It may also retrieve the appropriate representation in response to application requests, and support navigation among different representations (Vangenot et al. 2002). This work has been done within a European project, MurMur[1]. The approach has been validated using two test cases, a data management application and a risk assessment application, and has resulted in the development of software tools supporting data definition and query formulation for multiple representation of geographic databases.

[1] MurMur, Multiple Representation – Multiple Resolution, EEC 5th Framework project IST-1999-10723 (2000-2002). Project partners were IGN, Cemagref, Star Informatic, Université Libre de Bruxelles, University of Lausanne and Swiss Federal Institute of Technology, Lausanne. The project has been supported by EEC and the Swiss government (OFES). http://www.mur-mur.org

The next section delimits the scope of the paper, explaining the exact meaning we give to basic concepts such as multiple representation, consistency of representation and update propagation. Section 3 discusses the methodology that may be followed to move from a classical, mono-representation environment, to a multi-representation framework. Sections 4 and 5 develop the multi-representation modeling paradigm, discussing and illustrating how this applies at the schema and data levels, how different data structures may be defined to support multiple representation, how from the matching of instances that may exist in the source databases an appropriate schema can be defined to provide a unified multi-representation view of the existing data, and finally how different unified schemas may be designed according to application and designer preferred strategies. Section 6 uses a detailed example to illustrate the unification process, also showing the arguments that may be used to guide the process. Section 7 introduces related work, while the following section concludes the paper identifying possible directions for further work.

2. What is Multiple Representation

The database community used to stipulate that a database stores representations of those concrete and abstract things in the real world the applications are interested in. These representations are organized (structured and formatted) so that the data that is required by the applications can be stored in the database and later retrieved without ambiguity. Assume, for instance, a city management database is designed to support a variety of applications. The database stores data on roads within the city limits. A traffic management application considers roads as connections between two points. Its requirements may be translated by defining in the database schema[2] a RoadSection object type to represent, for instance, each road segment between two consecutive crossings. The RoadSection object type may be given properties, e.g., whether the road section permits movement only one-way or both ways, the number of lanes each way, and the average traffic density in each direction. Each instance of the object type is a (database) representation of a piece of reality that can be distinctively perceived when looking at the real world (e.g., the stretch of Broadway between the 42nd and 43rd streets).

[2] In this paper we adopt the database terminology, which refers to the description of data in a database as the database schema, and to the set of modeling concepts (object type, attribute, etc,) as the data model. In GIS applications the database schema is often referred to as the database model, and the data model is called the meta-model.

This representation is from the viewpoint of the traffic management application. The same piece of reality may also be of interest to a road maintenance service. However this service requires different data, e.g. the nature of the road pavement, when it was last repaired and the name of the enterprise having done the last repair. These properties define a different database representation, now from the viewpoint of the road maintenance service. In general, a database holds a number of representations to satisfy a variety of viewpoints, including multiple representations of the same piece of reality. We also say we have different representations when two instances are created in different types for the same real world entity, even if they hold the same data. For instance, there may be a representation of John Adams as an instance of the Person object type, and another representation as an instance of the Employee object type.

In geographical databases, different representations are often needed to cope with applications that require geometric features to exist at different resolution levels. Figure 1 illustrates an example of alternative geometric representations of the same piece of reality, a crossroad between two roads. The figure shows three possible representations for the same real world intersection shown on the photograph (the example is driven from the three databases that the French national mapping agency (IGN) currently maintains for its map production). Representation 2 is the less detailed (sufficient for 1:50'000 to 1:100'000 cartography) and depicts the crossroad as a node (defined as a common extremity of at least three lines) in the traffic network, where roads are geometrically represented as lines. The crossroad is simply the intersection point. Representation 3 supports car navigation applications (where scale varies between 1:10'000 in urban areas and 1:50'000 in countryside areas). It still represents the geometry of roads as lines, but uses a finer granularity that more precisely defines the traffic paths in the crossroad as a complex system of lines (here, the crossroad includes the circular road used to move around within the crossroad as well as the access road segments that may connect the crossing roads to the crossroad). Finally, representation 1 is used for topographic maps at a more detailed scale (1:5'000 to 1:50'000). It shows the geometry of the crossroad as a surface and requires high-resolution data acquisition. It is worth restating that in our approach (and in the user perception) the three geometric representations refer to the same real world crossroad, irrespective of the fact that one representation conveys more detail than the others. In the following, we show that the approach proposed here can be

used to maintain multi-representation semantics (i.e., the knowledge of what corresponds to what) whether the correspondence between instances materializing the same reality is a 1:1 mapping, a 1:n mapping or a n:m mapping.

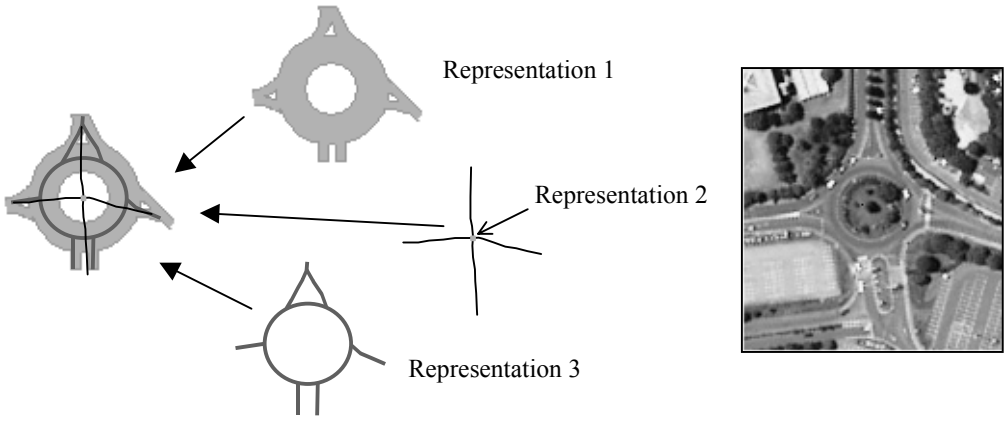

Figure 1: Alternative geometric representations of the same crossroad.

The left-hand side of Figure 1 shows how the representations may be geometrically superimposed in an integrated visualization. This illustrates a consistency requirement, which mandates that the geometries stored as part of the alternative representations should "match". Matching could be defined as ensuring that the centroid of representations 1 and 3 has the same coordinates as the intersection point in representation 2. Another consistency requirement, stated in terms of the current situation at IGN (i.e., the existence of three separate databases, one per application type) is that if a crossroad exists in the database holding representation 2, it should also exist in the databases holding representations 1 and 3. As no DBMS can currently handle inter-database consistency, IGN currently enforces these consistency rules by having updates trigger manual procedures. However in an integrated data set (such as the one we are targeting here) enforcement of these consistency rules would be the responsibility of the DBMS. Notice that consistency rules may be far more complex than the simple examples above. For instance, a rule could state "whenever there is an intersection between two lines representing two roads, if one of the roads is a highway and the other one is not a highway then there is no crossroad at the intersection point, instead there must be either a bridge or a tunnel". Thematic data is also subject to consistency rules.

Consistency rules and multi-representation semantics lead to the possibility of implementing update propagation. This consists in, whenever some data is updated, determining if additional updates are required to maintain data consistency and in this case automatically generating the additional update operations. For instance, if a crossroad is transformed from a simple intersection to a traffic circle, the creation of representation 1 for the crossroad could trigger the automatic creation of representation 3 based on some identified cartographic generalization rules that derive the line geometry from the surface geometry. If the creation of the traffic circle has moved the intersection point away from its original location, representation 2 should also be automatically updated to reflect the same change. The capability of propagating updates has a significant economic impact in reducing the cost of updating the integrated database and is the key to maintaining its consistency. Without update propagation and consistency rules, chances are high for an integrated database that represents a dynamic reality to rapidly become obsolete.

Generically speaking, differences in representation stem from differences in the intended use of data. Following common practice, we have referred to the latter as the "viewpoint", according to the metaphor that shows the database designer as "looking at" the world using a specific perspective for each different use intended for the database. The chosen viewpoint determines the suitable data structures and data values, as well as the desired spatial and temporal framework (in terms of extent and granularity). While the viewpoint concept can be taken as encompassing everything, in geographical applications it is often felt that resolution should be a separate factor that also influences representation. Indeed, users often feel like they want to have data available for the same viewpoint but at different resolution levels. It seems reasonable to adopt these two factors as independent dimensions in data representation:

- The viewpoint characterizes the way a specific application looks at the data. It filters what is of interest and how it is to be represented and stored, and what can be omitted, at the object (and relationship) type and instance level, as well as at the property level. For properties, it selects which attributes and methods have to be attached to each selected object and relationship type, and how attributes will be valued.
- The resolution determines the level of detail chosen to retain only relevant phenomena to be represented in the database (Steel and Worboys 1998). Geographic

resolution has to do with the minimal size of an extent that will be stored in the database. Semantic resolution determines the level of semantic detail in data acquisition (e.g., "cereal" versus "grain/corn/barley" are two ways to express land coverage at different levels of detail). The generic term resolution in this paper refers to both geographic and semantic resolution. Resolution also filters objects, relationships and properties.

In a multi-viewpoint and multi-resolution database, each representation holds for at least one viewpoint and one resolution. Actually, the concepts and the approach presented in this paper to deal with multiple representations do not depend on what criteria are used to characterize differences in representation (viewpoint, resolution, quality, time, etc., see (Hangouët 2003) for a discussion of the relevance of some of these criteria). We will nevertheless use viewpoint and resolution to illustrate our proposal.

3. From Multiple Databases to Multi-Representation Databases

An organization may have good reasons to set up multiple databases. At the IGN, these reasons included the following:

- IGN runs different applications with different focuses and goals. For instance, the road database focuses on connections between locations, while the topographic database focuses on topographic details and positioning. This requires alternative ways of grasping space (both in terms of resolution and viewpoint) and calls for different sets of reasoning processes and applications. Current DBMS technology makes it simpler to deal with this diversity by organizing separate databases.
- At the time the databases were set up, few derivation tools existed to perform automatic generalization of geographical data. Hence the solution to compute less detailed representations from more detailed ones was not available. This led to the development of separate products. Even nowadays, deriving an entire database by cartographic generalization is too weighty a task.
- It was anyway not possible to adopt a derivation strategy. Acquiring the detailed data and building the topographic database for the whole country has taken many years. Directly acquiring the derivable, lower resolution data went much faster than waiting for the detailed database to be completed.

Nevertheless, maintaining disjoint and independent databases nowadays has serious disadvantages for the IGN, as it complicates data management and induces costly update procedures (each change in the territory calls for three independent updating operations with their corresponding consistency controls).

The cost of maintaining consistency is not the only drawback. Many user applications require integrated access to multiple data sets that describe the same geographic extent at different resolutions. For instance, analysis and visualization tasks in some numeric cartography applications are improved when simultaneously using several levels of detail on the same extent. Other applications require data distributed over different products, e.g. using elevations together with street addresses requires a topographic database and a road/street database. In such cases, users have to match data (i.e., capture the correspondences between instances that represent the same real world entity) in order to be able to use different products together, a very complex operation (Harvey et al. 1998).
These considerations prompted the IGN to develop an interest in approaches leading to a multi-representation framework for its data. The process of building a common framework is called unification and is illustrated in Figure 2. The figure basically shows that no information is to be lost during the process.

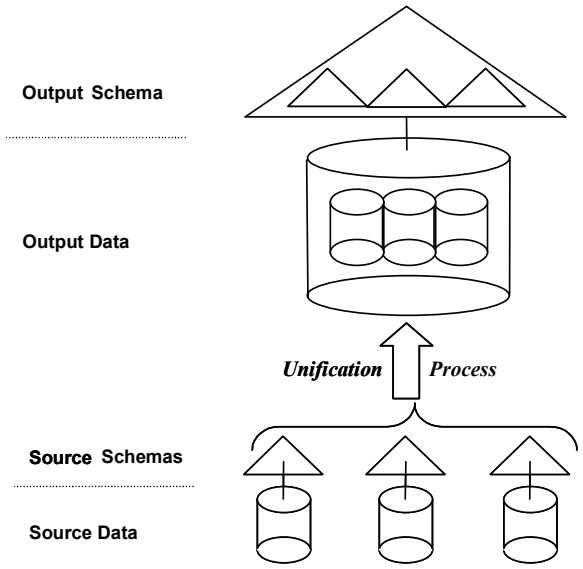

Figure 2: From several mono-representation databases to a single multi-representation database.

The data model used to define the output (global) schema has to have the ability to describe multiple representations. The output schema should support querying at the global level, thus enabling applications to simultaneously use multiple representations and navigate among them. It should also allow querying each representation independently, allowing legacy applications to keep running. Finally, it provides support for global consistency management.

The MurMur project provided the IGN with an opportunity to experiment the following steps towards a unification goal:

- **1) Choosing the approach.** Many approaches to unification in spatial databases are possible (Devogele et al. 1998). One approach, nowadays called the data warehousing approach, consists in replacing the initial databases with a new centralized database (Elmagarmid et al. 1999). Other approaches instead keep the existing databases and either build a centralized schema mapped to the existing ones (federated or mediation approach (Sheth and Larson 1990), (Parent and Spaccapietra 2000)), or just add relationships between existing schemas (Kilpeläinen 1998). Both the warehousing and the federated approaches require a multi-representation data model to describe the global vision. The MurMur project focused on developing such a data model and the associated schema definition and query formulation tools. To expedite testing, a centralized approach was assumed, which requires only a mapping to the underlying DBMS (no mediators needed).
- **2) Designing the global schema.** The IGN had to develop a new design for the global schema, choosing the most suitable data structures to model multi-representation situations. This task, a test case for the MurMur project, raised several opportunities to interact with the team developing the new data model to arrive at a data model definition consistent with user requirements.
- **3) Implementing the global schema.** The IGN participated in the specification and testing of the schema-editing tool developed by the MurMur project. Eventually, it used the tool to enter the global schema definition. The schema definition was automatically translated by the tool into an equivalent definition stored in an Oracle 9i database.
- **4) Data loading.** The IGN had to manually reverse engineer existing data to provide the data to be stored in the multi-representation database. This included using the

IGN's **data matching** algorithms and programs to identify correspondences between instances from the existing databases that described the same real world phenomenon.

- **5) Performing test queries.** The availability of software supporting multiple resolution databases. No DBMS or GIS has a logical data model that by default is able to support multi-representations. Hence, the multi-representation schema designed in step (2) needed to be transformed into a mono-representation schema, that can be implemented on a classic GIS or DBMS. To this purpose a software component was developed, with a user interface on one side, providing multi-representation functionalities, and with an existing system on the other side, implementing the multi-representation functionalities based on the existing functionalities of the underlying DBMS or GIS. This was done in the MurMur context, where a MADS (see below) to Oracle translation module was developed, enabling implementation of the multiple representations database in Oracle DBMS. To assess the adequacy of the design of the global schema, a number of test queries have been performed, using the visual query editor developed within the MurMur project. Queries are formulated in the MADS data model, and automatically translated for execution by the underlying Oracle DBMS. Results were provided using a data viewer also developed by the project.

Next sections focus on step (2), the definition of the data model, and show some examples from the use of the data model in the IGN test case.

4. Modeling Multiple Representations

This section briefly describes the MADS data model and the concepts that support multiple representations, and then we show in a small example the basics of the methodology to design the schema of a database with multiple representations. This methodology will be more thoroughly developed in the following sections, using examples from real IGN databases.

4.1 MADS and multiple representation concepts

When considering the need for multiple representations, it is easy to realize that anything in a database, from an object type to an attribute value, may require multiple

representations. An object type – as well as a relationship type – can bear different sets of properties according to the viewpoint in use. The characteristics of an attribute (its definition, its values) may also vary according to viewpoint and resolution. For instance, the set of possible values of an enumerated attribute, such as soil-cover, gets richer when semantic resolution is made more precise. The geometry attribute may hold different lines for the same road section according to the spatial resolution.

In other words, representation is a modeling dimension that is orthogonal to the data structure dimension. It can be added to any existing data model. In the MurMur project we have added multi-representation to MADS, a conceptual data model for spatio-temporal databases (Parent et al. 1999). MADS belongs to the family of entity relationship data models extended to support the main concepts of object orientation. It supports the following set of concepts:

– Object types, which can be organized into generalization hierarchies using is-a links. These hierarchies, contrary to the strict (and restrictive) purely object-oriented approach, can be dynamic and support multi-instantiation. Dynamic means that objects may change their membership in classes, moving from one object sub-class to another one. This allows representation, for instance, of the evolution of a road that is remodeled and consequently upgraded from state road to interstate road. In the database the object instance describing this road will move from the StateRoad object type to the InterstateRoad object type (while keeping its object identity). Multi-instantiation means that a unique real world entity may be described by two (or more) object instances belonging to any two classes in a hierarchy. For instance, a path section may at the same time be instantiated in the WalkingPath and in the BicyclePath object types, both sub-types of a Path super-type.
– Relationship types may link two or more object types. These may be cyclic, i.e., they may link an object type to itself. Like object types, they have a (relationship) identity, they may carry attributes and methods, and they may be organized into generalization hierarchies. Different kinds of relationship types are supported. A simple relationship type has no specific semantics or constraint. Aggregations are particular binary relationship types with the usual composition or part-of semantics. As in the real world many different kinds of aggregation can exist. No specific constraint is automatically attached to an aggregation. For instance, component objects may be shared by several composite objects. If this is not the case, the restriction may be

specified by the cardinality of the component role. Equivalence is another kind of binary relationship that has a specific meaning: "These two object instances represent the same real world entity". It is a symmetric relationship with an inherent constraint: Its cardinality is 1:1, which means that each instance of an object type linked by an equivalence relationship may be linked to at most one instance of the other object type. MADS also supports similar correspondences between two groups of instances (rather than individual instances). This kind of relationship is termed a set-to-set relationship type.

– Attributes may be simple or complex, i.e., composed of other attributes. They may be mono-valued or multi-valued: The value of a multi-valued attribute is a set (or list, or bag) of values.

To these data structuring capabilities, MADS orthogonally adds space and time modeling concepts.

– A set of spatial data types including the usual Point, Line and Surface simple types, plus composite ones, like PointSet, LineSet and ComplexSurface, and generic types such as Geo, the super-type of all spatial data types. These data types are equipped with associated methods. They are equivalent to those specified by the OpenGIS consortium (OpenGIS).
– In a similar way, MADS provides a set of temporal data types including the usual Instant and TimeInterval simple types, plus the composite types, InstantSet and IntervalSet, and a generic type, Time, that is the super-type of all temporal data types.
– In order to support a raster view of space as well as time-varying data, MADS offers a specific data type, Varying(D, S/T), that defines for any domain of values D, a new domain of values that varies in space, or time, or both. Each value of Varying(D, S/T) is a function that defines for each point in space (or each instant in time, or each (point, instant) pair) a value of D. Methods similar to those defined in (Güting 2000) are associated with Varying and allow users to query and manipulate these functions.
– Object types may be declared spatial, which means that they have a particular attribute, called geometry, which defines the spatial extent of the object. Its domain is one of the spatial data types. In the same way, a temporal object type has a particular attribute, called lifecycle, which defines the temporal extent of the object. Its domain is one of the temporal data types.

– Each object and relationship type may have any number of attributes with a thematic, spatial, temporal, or varying type.
– Finally, specific spatial (or temporal) relationship types convey a spatial (or temporal) semantics. These relationship types are binary. They link two spatial (or temporal) object types, and define a spatial (or temporal) constraint on the geometry (or lifecycle) attributes of the linked object instances. The constraint may be any of the usual topological (or Allen's interval) relationships, such as inside or before.

Figure 3 illustrates an example of diagrammatic notation used for object type definition in MADS.

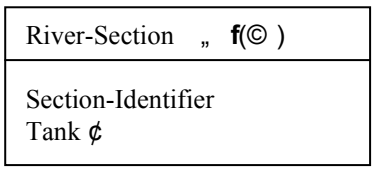

Figure 3: Example MADS diagram for a RiverSection object type.

The name of the object type is RiverSection. It has two attributes, Section-Identifier and Tank. The dot icon next to Tank shows that Tank is a spatial attribute of type Point. Similarly, the icon „ next to the name of the object type identifies the type as a spatial object type, therefore with a Geometry attribute. Its domain is OrientedLine, a sub-type of the Line data type. Next to the OrientedLine icon is another icon f(©) showing that the Geometry is a time-varying attribute: The course of a river may change during floods or after new embankments are built up, and the database will keep the history of these values.

The MADS spatio-temporal model has already been validated using various geographic applications. Relying on MADS made it easier to add representation modeling concepts by replicating the orthogonality-based approach we used for extending the model into the space and time dimensions. Let us call "representation scheme" the set of representations in the database that pertain to a given combination of values for the representation criteria in use. For instance, using the two criteria, viewpoint and resolution, a representation scheme, identified by (V_j, R_k), is the set of all representations holding for a given couple (V_j, R_k), where V_j is one of the supported viewpoints and R_k is one of the supported resolution levels. (V_j, R_k) examples for the IGN test case were: (Topographic, 1:10'000), (Topographic, 1:25'000), (Cartographic, 1:25'000), and (RoadNetwork, 1:50'000).

As a first step in extending the data model, we introduced a new data type, called DRstamp, to hold the set of allowed values denoting the supported representation schemes and provide the associated methods. Each database element is characterized by the representation schemes it belongs to. More intuitively, we say elements are representation-stamped, or simply stamped when there is no ambiguity concerning the nature of the stamp. Stamps materialize the representation scheme, they identify each element as a representation (or a part of a representation) valid for the corresponding representation scheme(s). Conversely, users accessing the database will specify the stamp(s) they work with (all stamps, a subset of the existing stamps, or a single specific stamp) and will consequently get all representations characterized by these stamps, and only these representations. More precisely, in a multi-representation database, each object (or relationship) type bears a particular multi-valued attribute, called, rstamps, of domain DRstamps. Its value is made up of the set of stamps for which the object (or relationship) type contains representations. Each attribute definition of an object or relationship type bearing several stamps can vary according to the representation scheme. Therefore, to each attribute definition can be associated a set of stamps that determine that this definition is valid for these representations. Finally, as each attribute value may also vary according to the representation scheme, the Varying data type has been extended to include representation-varying domains of values. Each representation is displayed in the schema with its associated stamp(s). An example is developed in Figure 4. To simplify the illustration, the figure only shows two stamps denoted as T and C.

4.2 Alternative modeling techniques for multi-representation

Equipped with the stamping mechanism and the rich set of relationship types supported by the MADS data model, the database designer is in a position to design a schema for a multi-representation database. Basically, there are two ways to organize multiple representations of the same phenomenon. One way is to put them all in a single container, e.g. an object type. Representation stamps are then used to qualify each element in the container so that it is known to which representation scheme(s) it belongs. This solution results in a real world entity being represented in the database as a single instance of an object type containing several representations, called a multi-representation object type. The other way, less explicit than the previous one, is to keep representations separate,

each one with its associated representation stamp, and link them with binary relationships whose semantics is "these are two representations of the same real world entity" (let us call this a correspondence relationship). In this solution, two representations of a same entity become two distinct instances of two different object types. These instances are linked by an instance of a correspondence relationship. There are different kinds of correspondence links, corresponding to different kinds of mappings between the representations: Is-a links, Equivalence, Aggregation and Set-to-Set relationships. Of course, the designer of a schema can choose any mix of the alternatives that best suits the requirements of the applications.

Figure 4 illustrates for a small example different solutions to restructure a set of independent databases describing (at least partially) the same real world to form a unified database storing the content of each initial database as a particular representation of the world. This constitutes one of theexisting goals of the IGN and is a frequent situation found in GIS applications. In this section we use a very limited example to focus on the proposed methodology. A more complete example is presented in Section 6.

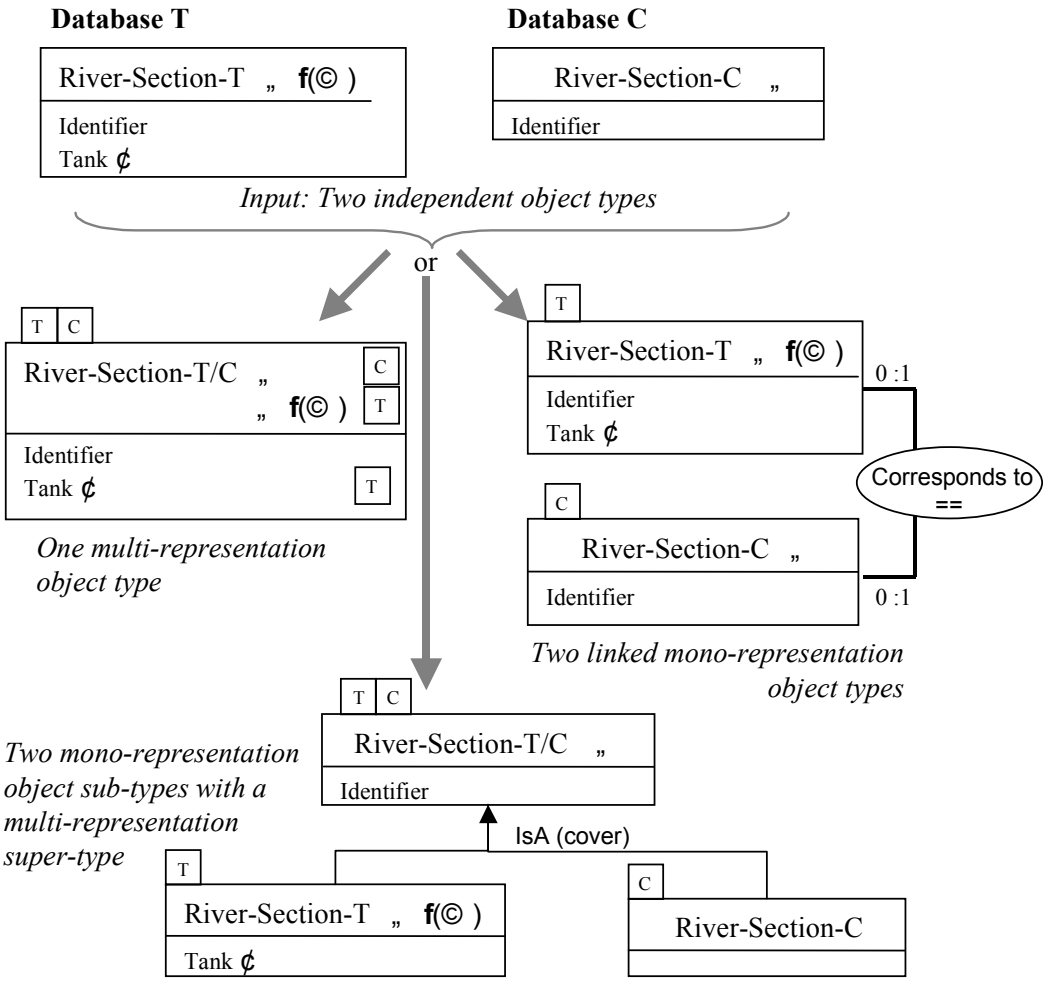

Figure 4: Possible descriptions of multiple representations

The initial situation, represented at the top of Figure 4, shows an excerpt of two geographic databases, T and C, describing the same country at two different resolutions: T is more precise than C. In each database a River-Section object type (River-Section-T, River-Section-C) describes the same rivers, segmented in sections. Some rivers are segmented in the same way in both databases. In this case, the Identifier attribute, which in both databases is made up of the name of the river followed by a sequence number identifying the segment, has the same value in corresponding object instances. Other rivers have not been segmented in the same way. In these cases, no match could be established between their segments. Therefore, globally speaking, the mapping between the instances of the two object types is a 0:1—0:1 mapping: For some instances a bijective mapping exists, for other instances no mapping exists.

Both River-Section object types have a Geometry attribute that represents the course of the river section by an oriented line. Database T records the history of the Geometry, as shown by the f(©) icon. On the other hand, database C records only the current value, at a less precise scale than database T. The two databases are consistent if the value in C is close to the value obtained by applying a cartographic generalization process to the last value of the geometry found in database T. The River-Section-T object type has an additional attribute, Tank. It is a spatial multi-valued and optional attribute of type Point (shown in the figure by a dot icon), which describes the locations of the tanks (if any) located on this river segment.

Assume now that databases T and C are merged into a global unified database. As part of the process, River-Section-T and River-Section-C will be remodeled to show that for some river segments the unified database holds two representations, a precise one like that supported by database T, and a coarser one like that within database C. The MADS data model with representation stamps allows us to design object types merging several representations and object types describing a unique representation but linked to other object types describing other representations. In this latter case, as the mapping between the representations describing the same real world river segment is 0:1—0:1, we can choose between Is-a links and equivalence relationships. Therefore we get the three different solutions that are illustrated in Figure 4.

– On the left, both representations are integrated into a single multi-representation object type. The object type bears two representation stamps (T and C), which indicate that it is visible to users working with either representation. The Identifier and Geometry attributes bear no stamp, which means that they have the same visibility as the object type. As the Geometry attribute integrates two different representations with different definitions (the T representation is precise and time-varying, the C representation is less precise and without history), two definitions appear in the schema, one for each representation. Two values will be stored in the database, one with history and one without history. The Tank attribute has only the T stamp. To gain access to the Tank attribute, users will have to specify the T stamp.
– On the right, the object types have been kept unchanged, but now they are stamped with their respective representation and linked to each other by a correspondence relationship. Attached to the link, an integrity constraint checks that, whenever two

objects are linked via this relationship, they have the same value for the Identifier attribute.

– The bottom of the figure illustrates another possible modeling solution mixing mono-representation and multi-representation object types. Data are organized into a generalization hierarchy, a modeling structure usually provided by object-oriented and semantic data models. The mono-representation object types are defined as sub-types of the multi-representation object type. The River-Section-T/C super-type gathers properties shared by both representations, T and C. In the example, this includes the Identifier attribute and the Geometry attribute, specified as in database C. The Identifier attribute is available in the two mono-representation sub-types, thanks to the inheritance rule embedded in the IsA link. The Geometry attribute is also inherited by the River-Section-C sub-type, while a new value (the more precise and time-varying one) is defined in the River-Section-T sub-type by overloading[3] the inherited Geometry of the super-type. As the Tank attribute is specific to representation C, it is added to the mono-representation River-Section-C sub-type.

It is up to the database designer to decide which data structure best fits his/her application requirements. The solutions are equivalent in the sense that they contain the same information. But the same query will be formulated differently depending on the chosen data structure. Hence, the designer should, as usual, evaluate which queries are the most frequent and chose the solution that makes those queries easier to write and faster to evaluate. For instance, if most queries access both representations, the left solution with a unique multi-representation object type is best suited.

5. From Instance Matching to the Definition of the Unified Schema

This section shows, for each kind of mapping between the object instances of the source databases that describe the same real world phenomena, the possible ways of modeling these multiple representations in a unified database. First, we need to determine the characteristics of the mapping. Therefore we have to build a match table for each pair of

[3] Overloading an attribute of a class is an object-oriented mechanism that defines in a sub-class a new value for this attribute. Whenever a user accesses the attribute, s/he gets the value defined in the super-class (respectively sub-class) if s/he is accessing the object instance as a member of the super-class (respectively sub-class).

source object types that contain instances that describe – as least partially and possibly with different viewpoints and resolutions – the same set of real world phenomena. The table will contain for each phenomenon the instances (or sets of instances) of the two object types that describe it. Examples of parts of match tables are given in Figure 5.

This search for matching representations relies on the properties that are shared, e.g., for the river segments of the previous section, the Identifier or Geometry attributes. The matching may be done manually by looking at the data, for instance by superimposing the maps described by the source databases. But manual matching is feasible only with toy databases. With real databases, a matching rule may exist that defines the mapping between the instances of two object types. In the river segment example, this rule could be: "Match instances of River-Section-T with instances of River-Section-C whenever they share the same value for their Identifier attributes". This is an example of thematic, or semantic matching. Frequently, the matching rule complements a thematic matching with a spatial matching of the values of spatial attributes (Lemarié and Raynal 1996) (Harvey et al. 1998). Even in the case where a thematic criterion completely defines the mapping, the representations may share other, possibly spatial, attributes whose matching must also be checked to guarantee the consistency of the unified data. In the river example, this means that, once two matching river sections have been found, based on the thematic criterion (same value for Identifier), one still has to check that the value of Geometry in River-Section-C is approximately corresponding (possibly through cartographic generalization) to one of the most recent values of Geometry in River-Section-T. If a matching rule can be defined, then a program can apply it to build the match table. When it is difficult to find and precisely express a matching rule, learning algorithms may be used to automatically generate the match table without explicitly defining any matching rule (Badard 1999) (Bel Hadj and Vauglin 1999).

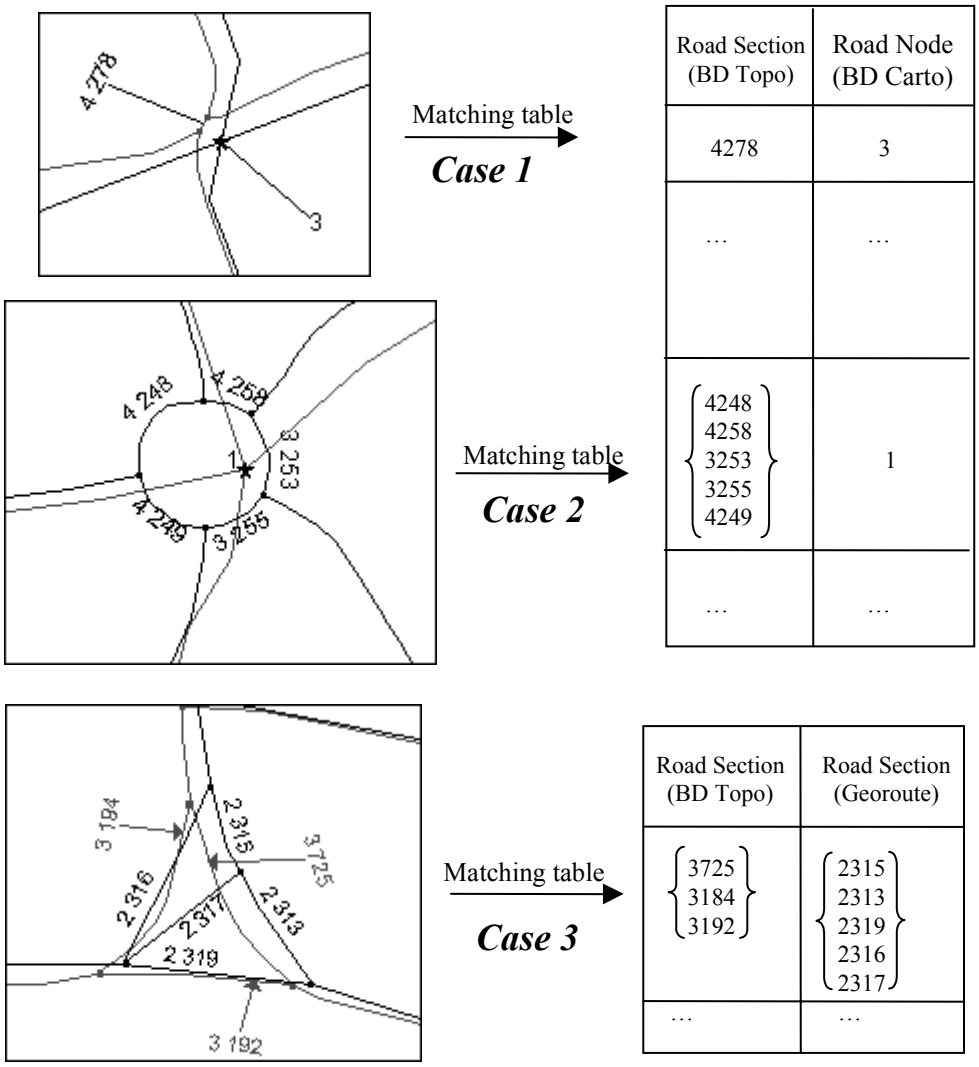

Figure 5: Examples of spatial matching and match tables.

Figure 5 illustrates three examples of spatial matching of crossroads between the three IGN databases, BDTopo, BDCarto, and Georoute. In BDCarto, the spatial object type, Road-Node, whose Geometry is of type Point, describes different kinds of nodes, including the crossroads. In the other two databases, BDTopo and Georoute, crossroads are not described per se, but through their component road segments. Both databases contain an object type called Road-Section, whose Geometry is of type Line, and which describes all road segments. Some instances of both Road-Section object types describe road segments that make up crossroads. Let us see for a few real examples, the mappings between these three object types. The first two examples in Figure 5 show how two crossroads are

described in BDTopo and BDCarto. The last example shows a crossroads as described in BDTopo and Georoute.

- Mapping crossroads vetween BDTopo and BDCarto: Between Road-Section of BDTopo and Road-Node of BDCarto, one may find either a 1—1 matching (case 1 of Figure 5: one instance of Road-Section corresponds to one instance of Road-Node), or a 1—N matching (case 2: one instance of Road-Node corresponds to a set of instances of Road-Section). Notice that while the geometry from N instances of Road-Section (a set of lines) can be generalized into a single point (the geometry of a Road-Node), the inverse is not true. So there is no N—M mapping between the populations of the two object types. Both the 1—1 and 1—N matching correspond to correct situations and the database should be able to store data accordingly.
- Mapping crossroads between BDTopo and Georoute: N—M mappings between instances of Road-Section of BDTopo and instances of Road-Section of Georoute are likely to happen frequently. An example is shown in case 3 of Figure 5: three BDTopo road segments correspond to five road segments of Georoute. Notice that segment 2317 is included in the mapping as it is part of this crossroads in Georoute.

The result of building the match tables defines the crossroads mapping between BDTopo and BDCarto as 1—1 or 1—N, depending on the crossroads, while the mapping between BDTopo and Georoute is always N—M. At this point the designer of the unified database can choose a suitable data structure for each case. Possible MADS structures are:

- A 1—1 mapping can be described by an object type grouping all representations, or two object types linked by an IsA link, if the populations of the two object types satisfy the inclusion property, or otherwise by an Equivalence relationship. This was the case described in Figure 4 for river segments.
- A 1—N mapping can be described by two mono-representation object types linked by an Aggregation relationship.
- A N—M mapping is a bit different. The real world phenomenon that is represented by the mapping is not explicitly described in any of the two databases. The designer should first check if this phenomenon is explicitly described in another source database, e.g., by an object type. That is the case in our example: Crossroads are described in BDCarto as instances of Road-Node. In this case, the N—M mapping can be split into two 1—N mappings. If the phenomenon is not described in any source as a whole, then, if it is interesting for the future applications to get access to these

phenomena as whole objects, a new object type is added to the unified schema with two Aggregation relationships. It is the same solution as the previous one. If having the phenomena explicitly described is not of interest, the phenomena remain implicit and the mapping is described by two mono-representation object types linked by a Set-to-Set relationship.

Let us now continue the crossroads example to show a few possible MADS schemas that describe the unified database for crossroads. According to application needs, the designer may choose to go for a loosely integrated schema, showing all source object types without any modification, or, on the contrary, a more concise schema that merges as much as possible the corresponding object types into a unique multi-representation object type.

5.1 A loosely integrated schema

If the application needs to keep in the unified database all the source object types exactly as they were in the source databases, and wants only to interrelate the corresponding objects by correspondence links, the unified schema will resemble the one shown in Figure 6. In this schema, each source object type is kept unchanged and put in the unified schema as a mono-representation object type (as shown by the unique representation stamp at the top of the box corresponding to each object type). Correspondence links have been set up in between the object types. They explicitly describe the mapping defined by the match tables from Figure 5. A program will generate, for each line of the match tables, the corresponding relationship instance linking the object instances. The object types Road-Section-T (from BDTopo) and Road-Section-G (from Georoute) are linked by a Set-to-Set relationship which is the only relationship that can express an N—M mapping. For the object types Road-Section-T and Road-Node-C (from BDCarto) the mapping depends on the real world crossroads represented and is either 1—1 or 1—N. The simplest modeling solution is to describe the mapping by an Aggregation relationship, which can describe both kinds of mappings. But for IGN applications, the case of crossroads represented by a unique road segment in BDTopo is important and should be explicitly described in the unified schema. Therefore, an Aggregation represents the generic 1—N cases, and an Equivalence relationship, which is a sub-type of the Aggregation relationship, represents the 1—1 matching cases. The benefit of this

design is that whoever looks at the unified schema immediately perceives the fact that the mapping may have different cardinalities at the instance level.

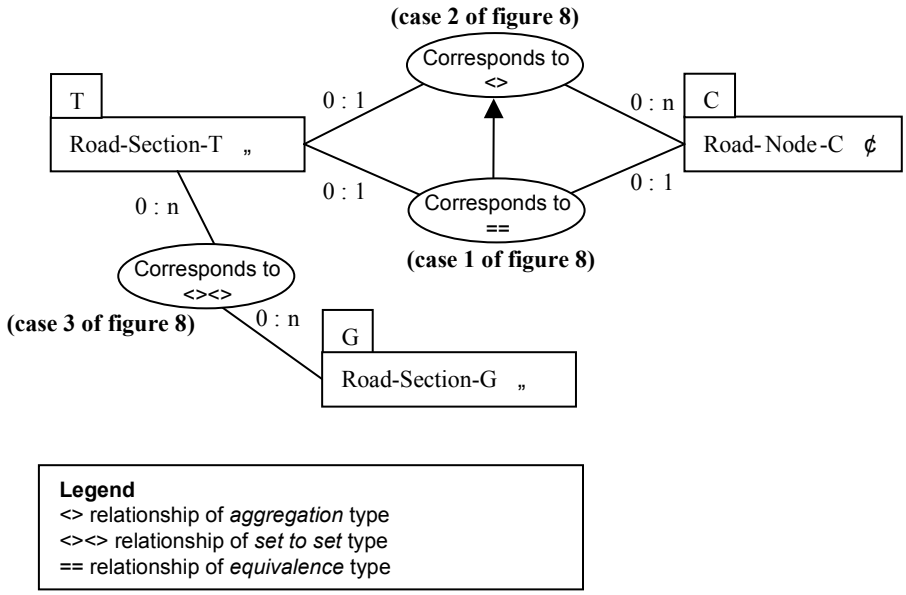

Figure 6: Correspondence description in a loosely integrated design.

5.2 A more integrated schema

Let us now assume that the application requires a unified schema that concisely and explicitly describes the real world phenomena that are – at least partly – represented in several source databases. A preferred strategy is to merge all corresponding source object types into a unique multi-representation object type. But this solution cannot be used because the cardinalities of some correspondence links are not 1—1. In fact, a 1—N (or N—M) correspondence link means that the linked representation(s) on the N (and M) side do not describe the whole real world entity, but different parts of it. It would not be meaningful to merge within the same object type, at the same level, representations that do not describe the same phenomenon at that level. Indeed, other solutions may be designed by adding new object types in the unified schema.

For instance, the solution illustrated in Figure 7 consists in creating a new multi-representation object type to explicitly represent the concept of a crossroads. This object type bears the three stamps (from the three source databases), and is linked by

correspondence relationship types to the three mono-representation object types coming from the source databases. Road-Node-C and Crossroads are linked by an Equivalence relationship, as the correspondence link between the two is 1—1. Road-Section-T and Road-Section-G are both linked to Crossroads through Aggregation relationships, as a real crossroads is represented in BDTopo as well as in Georoute by a set of road segments.

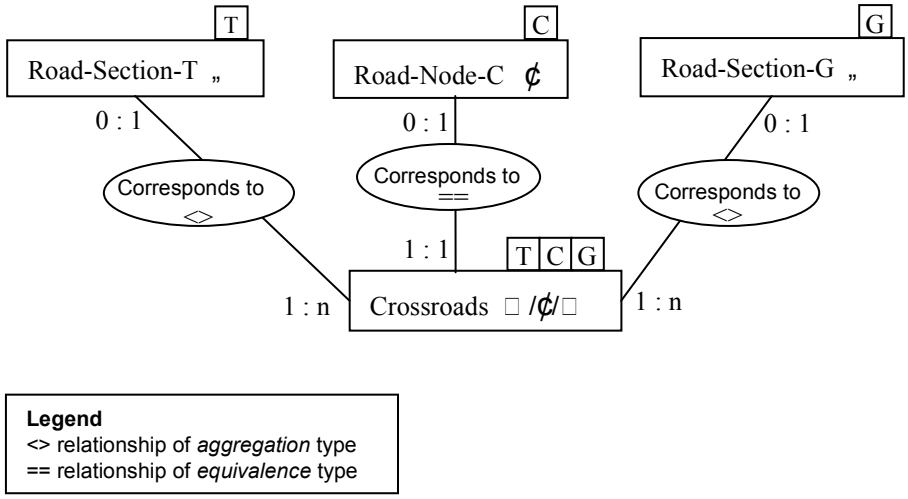

Figure 7: An integrated design with mono- and multi-representation object types.

6. An Example: Crossroads and Tollgates at the IGN

This section illustrates how the IGN used the multi-representation concepts to build up a unified database merging the three source databases, BDTopo, BDCarto, and GeoRoute. We will not explain the whole process, we will focus on the design of a part of the unified schema: the part that describes crossroads and tollgates. Figure 8 shows the relevant input object types in the three source databases:

– BDCarto splits crossroads into two categories. A crossroads that is less than 100 meters wide, whatever its complexity, is represented by a single object in Road-Node-C, whose geometry is of type Point. For each crossroads wider than 100 meters, several elementary nodes are stored in Road-Node-C, and an instance of Complex-Crossroads-C (whose geometry is of type Set-of-Points) is created as the composition of these nodes. The value of the Kind attribute of Complex-Crossroads-C may be “traffic circle”, “motorway intersection”, or “made-up-crossroads”. This last value, “made-

up-crossroads", denotes complex crossroads equipped with constructions separating the lanes.

– In BDTopo, every crossroads (simple or complex), up to 25 meters wide, and wherever the traffic is structured – i.e. where lanes are distinct – is seen as a node of the road network and stored as an object in the NM-Crossroads-T (non-made-up-crossroads) object type. NM-Crossroads-T has geometry of type Point. Complex crossroads wider than 25 meters are decomposed into independent nodes, and these nodes are stored as NM-Crossroads-T objects too. Another object type, Crossroads-T, with geometry of type Surface, stores the unstructured crossroads (large squares where roads meet without any delimited traffic lane).
– In the Georoute database, simple and complex crossroads, irrespectively of their size, are stored in the Road-Node-G object type, which has geometry of type Point. Unstructured crossroads, made up crossroads, and traffic circles that are more than 30 meters wide are also represented as instances of the Complex-Crossroads-G object type. The Complex-Crossroads-G object type has a geometry of type Surface, and the value of its attribute Kind-of-crossroads is "traffic circle", "made up crossroads", or "unstructured traffic zone (ZTNS)".
–

Examples of possible values for the attributes Type (of Road-Node-C and NM-Crossroads-T) and Kind-of-node (of Road-Node-G) include "simple crossroads", "change of traffic conditions", "barrier", and "traffic circle". These three object types do not contain the same set of instances: There are more crossroads represented in the Georoute and the BDTopo types than in the BDCarto type, because of the coarser resolution of BDCarto.

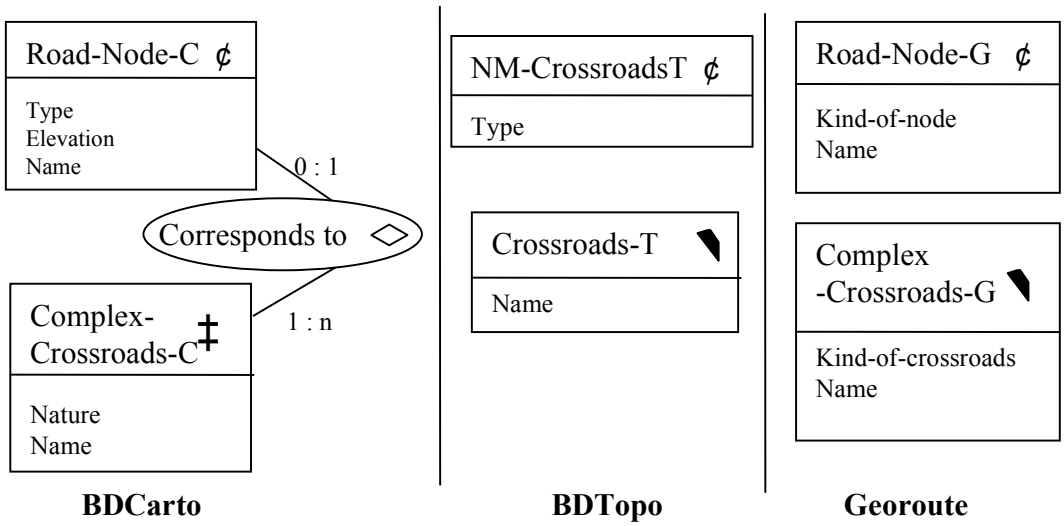

Figure 8: Representation of crossroads in three source databases.

The goal of the IGN during the MurMur project was to merge the three source databases into a global database in order to improve the global consistency of the data and to set up a structure that will be able to support the automatic propagation of the updates. The representations of crossroads in the three databases are very different: Different threshold sizes separating simple crossroads from complex ones, different types of geometry, different sets of real world entities represented. Such a variety makes it difficult to merge several source object types into multi-representation object types. Therefore a loosely integrated design was chosen: all the object types from the three source databases have been put into the unified schema as mono-representation object types, and correspondence links have been established between them. Indeed, consistency checking and update propagation do not require a more integrated database; hence a loosely integrated database is sufficiently expressive.

As the unified schema is large, it has been split in two parts, shown in Figures 9a and 9b.

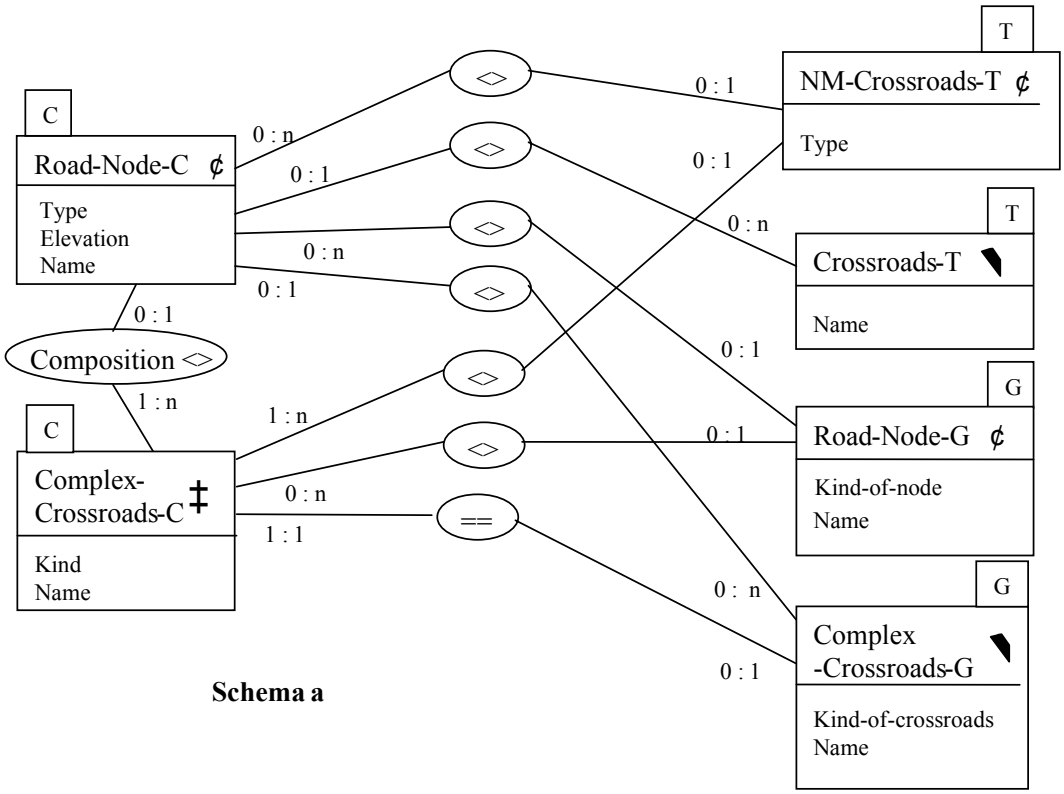

Figure 9a: The first unified sub-schema for crossroads.

The apparent complexity of the schema is due to the goal of making visible, as precisely as possible, the characteristics of the mappings between the instances of corresponding object types. For example, Figure 9a shows an Equivalence relationship type between Complex-Crossroads-C and Complex-Crossroads-G. The choice of an Equivalence relationship is inferred from the definitions of both object types: a Complex-Crossroads-C is wider than 100m and hence is always a Complex-Crossroads-G (wider than 30m). This explains the 1:1 cardinality of the role of Complex-Crossroads-C. On the other hand, a Complex-Crossroads-G that is larger than 30m but smaller than 100m will not be stored in Complex-Crossroads-C. So the cardinality of the role of Complex-Crossroads-G is 0:1.

As the Georoute and BDTopo databases have similar resolutions, they contain approximately the same number of simple road nodes. Therefore, in Figure 9b, the link between NM-Crossroads-T and Road-Node-G is a Set-to-Set relationship. The link between Complex-Crossroads-G and Crossroads-T exists only if the Complex-Crossroads-G has the type ZTNS. In this case, a corresponding instance does necessarily exist in Crossroads-T,

since there is no size condition for this object type. This is the reason for the 1:1 cardinality on that link.

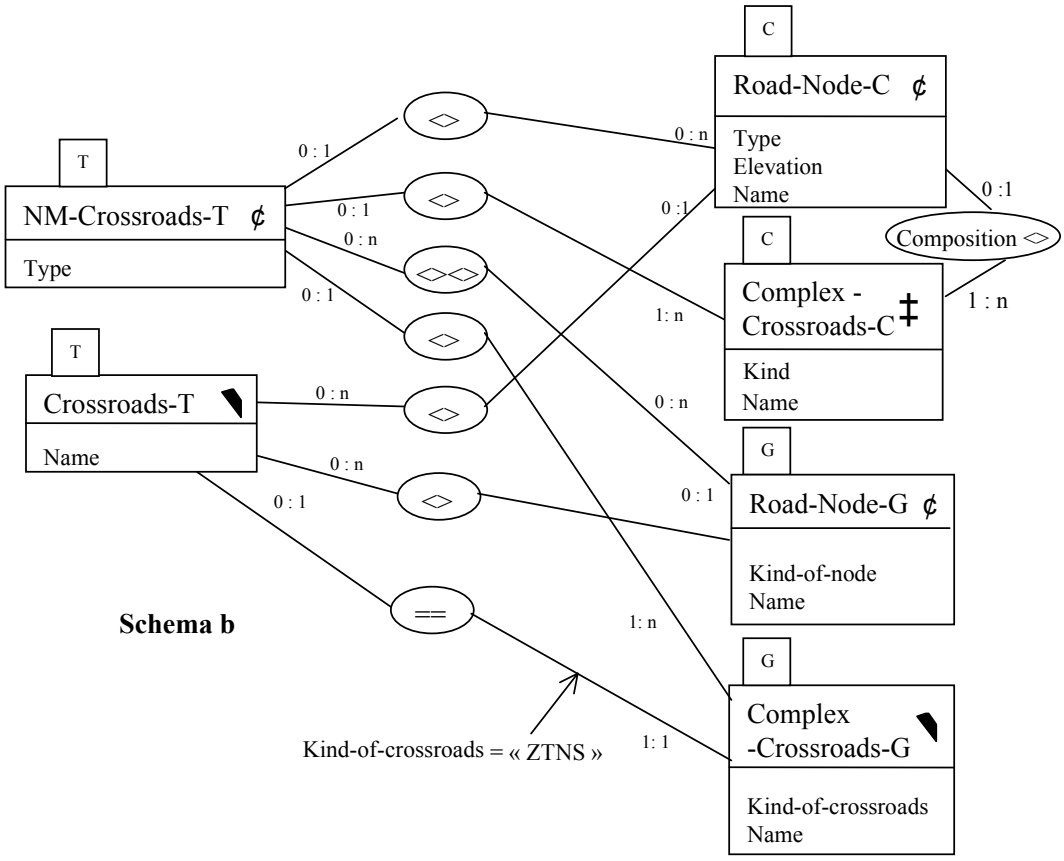

Figure 9b: The second unified sub-schema for crossroads.

Figure 10 provides another part of the unified schema that also uses correspondence links to correlate mono-representation object types. The schema describes tolls on the road network, which are represented differently in the three IGN databases. In BDTopo, a tollgate is a specific object type, with a Surface geometry, symbolizing the building where traffic fees are paid. In BDCarto, tollgates belong to the Road-Facility-C object type that also describes rest areas, service areas, and small tunnels. Its geometric type is Point. Similarly, tollgates in Georoute are represented by a subset of the instances of the Road-Node-G object type. They differ from other road nodes on the Kind-of-node attribute value.

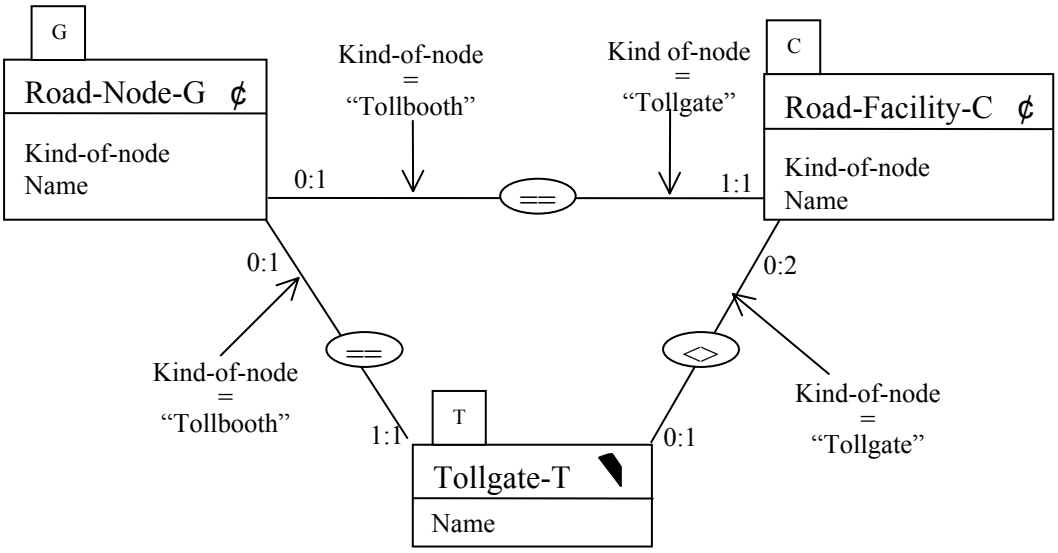

Figure 10: Tolls on the road network as correlated instances of mono-representation types.

In Figure 10, as in the previous figure, the arrows pointing to roles of relationships describe integrity constraints that are specific to the multi-representation. For example, the indication "Kind-of-node = «Tollgate»" between Road-Facility-C and Tollgate-T means that only instances of Road-Facility-C where the value of the attribute Kind-of-node is equal to "Tollgate" participate in relationships with objects of Tollgate-T. If this condition is not met, this is considered as an integrity constraint violation.

Each source database is *a priori* considered to be consistent with regard to its specifications. But unifying a set of databases together into a multi-representation database leads to conflicts between representations. Adding integrity constraints to the target schema, as in Figure 10, ensures that only consistent representations are stored in the unified database. This technique helps to identify and solve most of the conflicts. Nonetheless, some topological conflicts and semantics errors may remain, because there is no simple integrity constraint that may filter them out. Solutions for automatically determining the type of conflict and solving these more accurately are currently under investigation at the IGN's COGIT laboratory.

7. Related Work

To the best of our knowledge, the approach reported in this paper is the first attempt to address multi-representation as a generic modeling issue and as an additional modeling dimension. Conversely, there has been a large amount of work addressing specific facets of multi-representation.

The most popular technique supporting multiple representations in databases is the view mechanism. Views in relational DBMSs provide great flexibility in rearranging underlying basic data items into new structures that either express alternative representations of the entities described in the database (views defined by selections and projections), or define new entities (views defined by restructuring operations like joins). As views are derived from the basic tables, views are not a mechanism to introduce new representations that have not been planned and integrated beforehand in the structures of the basic schema. Also, in a view framework there is no concept of a representation scheme, as defined in this paper. This means that there is no schema that shows how representations are organized (for instance, who is using what) and one has to resort to access right definitions to find out which view may form a consistent representation of the world.

Object-oriented DBMS provide more limited support for views. On the other hand, they improve on relational systems by supporting the IsA construct, thus allowing multiple classifications (one facet of multi-representation). However, this flexibility is strongly limited by two facts. The generalization hierarchy is mono-instantiation. Hence each real world entity may be represented by at most one most specialized instance in the hierarchy. Furthermore, the hierarchy is static. An instance can never move from one class to another one. This has prompted numerous proposals to eliminate this limitation, basically supporting the "role" concept as a way to define additional classes where a given real world object can be instantiated (see, e.g., (Papazoglou et al. 1994)). However, these role approaches can only provide alternative representations for 1:1 or 0:1 matching situations, while our approach also covers 1:N and N:M mappings between representations. Versioning (Peerbocus et al. 2002) and temporal database techniques are other examples where a specific facet of multi-representation is supported.

Level of detail aspects in multi-representation have been mainly addressed by research in spatial databases, namely multi-scale databases, i.e., databases that allow an object to be characterized by several representations of its geometry, each one at a different scale. Because of the largely hierarchical nature of transitions between scales, multiple representations in multi-scale databases are most often organized into hierarchical data structures, where levels in the hierarchy correspond to increasing detail (Timpf and Franck 1995) (Timpf 1998) (Kilpeläinen 1998) (Zhou and Jones 2001).

A structure suitable for processing and reasoning with spatial data sets that are heterogeneous with regard to semantic and spatial resolution is proposed in (Steel and Worboys 1998). This structure, called *stratified map space*, consists of a granularity lattice and a *map space* gathering schemas and data sets that correspond to a given semantic granularity. Compared to this research in spatial databases, we mainly differ in that we address multi-representation in general, i.e., not limited to the geometry of objects, and we do not restrict the approach to hierarchical structures of the set of representations.

Finally, a special note should be made to the work of Bédard and his team on the Vuel concept (Bédard and Bernier 2002). This work has very similar motivations to our work, but takes a different path to multi-representation. As the name sugests, the Vuel is an approach that obeys the traditional view approach. A Vuel is a view complemented with spatio-temporal and presentation features. The approach shows the same advantages and disadvantages as relational views.

8. Conclusion and Future Work

Geographical data strongly depend on both point of view and the user's needs, which lead to several representations of the same geographical phenomena. Having different representations available means richer information. This can be made systematically and consistently available to users by performing data matching once and providing matched data to users in a multi-representation system.

A framework for modeling several representations in geographical information systems and creating a multi-representation database from existing databases has been described,

and illustrated with examples. It provides an innovative solution that has not been explored before. The paper discussed alternative approaches that allow adapting the design of the multi-representation database to the data management strategy of the organization. Examples from IGN applications using multiple representations have been provided. The MurMur project that performed this work also successfully ran a second test case on a natural risk management application with the goal of defining a strongly integrated database.

Beyond the features and developments described in this paper, additional issues remain to be investigated to gain further benefits from multiple representations within GIS. Among them the IGN has already identified the following actions:

- Develop strategies and techniques to avoid redundant data capture.
- Manage the inconsistencies between representations. This means that conflicts between representations would not be left to the user, but data producers would solve them.
- Derive new products. This requires further research on automatic cartographic generalization.
-

For the general benefit of the GIS community, we plan to address the following research issues:

- Selection of information used on queries: How to compute a distance, a position, a density, etc in a multi-representation database? Results differ if one representation is used instead of another. This question has been partially addressed by (Claramunt and Mainguenaud 1996), but it remains open for geographical databases.
- Re-engineering methods and tools for extracting from an existing geographic data source an explicit and conceptual description of its structure and content, associated with a mapping to the source data. This would greatly help to reuse the large amount of exiting geographic data.

This paper has shown that even with the simplest means of modeling multiple representations (linking mono-representation object types through correspondence relationships), a significant potential for use is provided without being too complicated. We strongly believe that this trend will be developed in the next years.

Acknowledgement

The authors wish to express their thanks to François Vauglin who initiated work on this paper before he left the IGN and the MurMur project. We also thank all the MurMur partners.

References

ANZLIC, 1996: ANZLIC, Spatial Data Infrastructure for Australia and New Zealand, November 1996. http://www.anzlic.org.au/asdi/anzdiscu.htm

Badard, T., 1999, On the automatic retrieval of updates in geographic databases based on geographical data matching tools. In *Proceedings of the 19th International Cartographic Conference ICA'99* (Ottawa, Canada), pp.47-56.

Bédard, Y., and Bernier, E., 2002. Supporting Multiple Representations with Spatial View Management and the Concept of "VUEL". In *Proceedings of the Joint Workshop on Multi-Scale Representations of Spatial Data, ISPRS WG IV/3, ICA Com. on Map Generalization* (Ottawa, Canada), July 7th-8th.

Bel Hadj Ali, A., and Vauglin, F., 1999, Geometrical matching of polygons in GIS and assessment of geometrical quality of polygons. In *Proceedings of the 2nd International Symposium on Spatial Data Quality* (Hong Kong), pp.33-43.

Claramunt, C., and Mainguenaud, M., 1996, A spatial data model for navigation knowledge. In *Proceedings of 7th International symposium on Spatial Data Handling SDH'96* (Delft, The Netherlands), pp. 12A31-12A48.

Devogele, T., Parent, C., and Spaccapietra, S., 1998, On spatial database integration. *International Journal of Geographical Information Science*, Vol.12, N°4, pp. 335-352.

Elamgarmid, A., Rusinkiewicz, M., and Sheth, A., 1999, *Management of Heterogeneous and Autonomous Database Systems* (Morgan Kaufmann).

FME. Feature Manipulation Engine, Safe Software Inc., http://www.safe.com/products/fme/index.htm

FGDC, 1997. Federal Geographical Data Committee, A Strategy for the National Data Infrastructure. April 1997, http://www.fgdc.gov/nsdi/strategy/index.html

GeoConnections, 2001. GeoConnections Policy Node: Proposed Canadian Government Action Plan On Geospatial Action Policy, October 1997, http://cgdi.gc.ca/docsIndex.cfm?lang=en

Guarino, N., 1998. Formal Ontology and Information Systems. In *Proceedings of the First International Conference on Formal Ontologies in Information Systems*, FOIS'98, (Trento, Italy) IOS Press, June, pp. 3-15

Güting, R.H., Böhlen, M.H., Erwig, M., Jensen, C.S., Lorentzos, N.A., Schneider, M., and Vazirgiannis, M., 2000, A Foundation for Representing and Querying Moving Objects, *ACM Transactions on Database Systems,* Vol.25 N°1, pp.1-42.

Hangouët, J.F, 2003, Geographical multi-representation: Striving for the hyphenation. *International Journal of Geographical Information Systems, Data Fusion Special Issue.*

Harvey, F., Vauglin, F., and Bel Hadj Ali, A., 1998, Geometric matching of areas, comparison measures and association links. In *Proceedings of the 8th International Symposium On Spatial Data Handling* (Vancouver, Canada), pp.557-568.

Kilpeläinen, T., 1998, Maintenance of topographic data by multiple representations. In *Proceedings of the Annual Conference and Exposition of GIS/LIS'98* (Forth Worth, Texas), pp. 342-351

INSPIRE, INfrastructure for SPatial InfoRmation in Europe, http://inspire.jrc.it/home.html

Lemarié, C., and Raynal, L., 1996, Geographic data matching: first investigation for a generic tool. In *Proceeding Annual Conference and Exposition of GIS/LIS'96* (Denver, Colorado), pp. 405-420.

OpenGIS, Open GIS Consortium, Inc, http://www.opengis.org/

Papazoglou, M.P., Kramer, B.J., and Bouguettaya, A., 1994, On the representation of objects with polymorphic shape and behavior. In *Proceedings of the 13th International Conference on Entity-Relationship Approach* (Manchester), pp.223-240.

Parent, C., Spaccapietra, S., and Zimanyi, E., 1999, Spatio-temporal conceptual models: Data structure + space + time. In *Proceedings of 7th ACM symposium on geographic information systems, GIS'99* (Kansas City, Missouri), pp.26-33.

Parent, C., and Spaccapietra, S., 2000, Database Integration: The Key to Data Interoperability. In *Advances in Object-Oriented Data Modeling*, edited by M.P.Papazoglou, S.Spaccapietra, and Z.Tari, (The MIT Press), pp.221-253.

Peerbocus, M.A, Jomier, G., and Badard, T., 2002, A Methodology for Updating Geographic Databases Using Map Versions. In *Proceedings of Spatial Data Handling Conference* (Ottawa), pp.363-376.

Sheth, A., and Larson, J., 1990, Federated database systems for managing distributed, heterogeneous, and autonomous databases, *ACM Computing Surveys*, Vol.22, N°3, pp. 183-236.

Steel, J., and Worboys, M., 1998, Stratified map space: A formal basis for multi-resolution spatial databases. In *Proceedings of the 8th Symposium on Spatial Data Handling, SDH'98*, pp.180-189.

Timpf, S., and Franck, A., 1995, A multi-scale DAG for cartographic objects. In *Proceedings of Auto-Carto 12* (Charlotte, North Caroline, USA) pp.157-163.

Timpf, S., 1998, Hierarchical structures in map series. Phd Thesis, Technical University of Vienna.

Vangenot, C., Parent, C., and Spaccapietra, S., 2002, Modeling and Manipulating Multiple Representations of Spatial Data. In *Proceedings of Spatial Data Handling Conference SDH'02* (Ottawa), pp.81-93.

Ye, X., Parent, C., and Spaccapietra, S., 1997, View Definition and Positioning in DOOD Systems. *Journal of Systems Integration,* Vol.7, N°3/4, pp.263-290.

Wiederhold, G., 1992, Mediators in the Architecture of Future Information Systems *IEEE Computer*, Vol.25, N°3, pp. 38-49.

Zhou, S., and Jones, C.B., 2001, Design and Implementation of Multi-Scale Spatial Databases. In *procedings of the 7th International Symposium on Spatial and Temporal Databases* (Redondo Beach, CA).