Link between the Whole and its Parts in UML Representations of Spatial Aggregations: An Application in the Context of Geographic Databases

François Pinet[1], Myoung-Ah Kang[2], Hélène Jaudoin[1]
[1] Cemagref Clermont Ferrand, France
{francois.pinet, helene.jaudoin}@cemagref.fr
[2] Laboratory of Computer Science, Modelling and System Optimisation (LIMOS)
ISIMA / Clermont Ferrand University, France
kang@isima.fr

1. INTRODUCTION

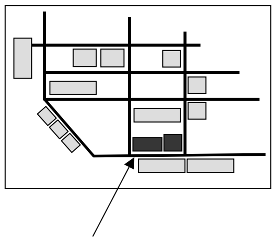
1.1 Context

Extended relational models used for geographic databases allow to store spatial data. Indeed, the common extension consists in adding a special attribute in tables; the values of this attribute in database instances are collections of geometries. For example, this type of structure is implemented in the Geographic Information System (GIS) MapInfo (MapInfo Corporation, 2003). Figure 1 describes an example of this relational model extension: in the table Building, the value of the attribute geometry is a collection that contains several simple polygons (one polygon for each part of a building).

Building
building_id building_name geometry

a. Structure of a table Building described in using the UML formalism

building_id	building_name
112	Pascal A
113	Pascal B
120	Einsten A
121	Einsten B
122	Einsten C
123	Socrate A
124	Socrate B
130	Aristote
131	Platon A
132	Platon B
133	Platon C
134	Platon D
135	Thales A
136	Thales B



Each part of the building 124 is selected on the map

b. Instances of the table Building (alphanumeric features are presented in a table and the value of the attribute geometry appears on a map). The instance corresponding to the building 124 is selected; this building is divided into two parts (two simple polygons).

Figure 1: Implementation of a table Building in a geographic database and example of instances. The features of a building are composed of three attributes (building_id, building_name and geometry) and the geometry attribute has a cartographic representation.

Furthermore, the aggregation of spatial data is an important operation used in geographic databases (Kösters, 1997; Tryfona, 1997). It consists in grouping inside a geometry attribute value, several geometries initially stored in different instances of diverse tables.

At a conceptual level, spatial aggregation operations are usually modelled by database designers thanks the Unified Modeling Language (UML) (Booch, 1998; Bédard, 1999; Borges, 1999; Brodeur, 2000; Jugurta, 1999; Lbath, 2000; Laurini, 2001). In this type of modeling, the semantics of the UML aggregation are extended in order to define a dependancy between geometry attribute values of whole classes and geometry attribute values of part classes.

Figure 2 shows a spatial aggregation defined with UML notations; indeed, in the presented class diagram, aggregation relationships are drawn between wholes and their parts. In this example, each class of the UML diagram represents a table of a geographic database and the relationships linking the classes indicate how the wholes are computed from the parts. The goal of the presented aggregation is to create:

- a Country table from Capital_Town_Representation1, Capital_Town_Representation2, Frontier and Road_Section tables,
- then, a Continent table from the Country table.

Capital_Town_Representation1, Capital_Town_Representation2, Frontier and Road_Section are input tables and the output tables are Country and Continent. In the conceptual model of the example, spatial types of geometries stored in input tables are specified in using the UML remark notation. Thus, a country is composed of a frontier (a polygon), zero, one or many road sections (polylines), and a capital town (represented by either a point or a polygon but not both). In the same way, the diagram indicates that a continent is made up of several countries. According to the UML notation, a xor constraint can link two aggregation relationships.

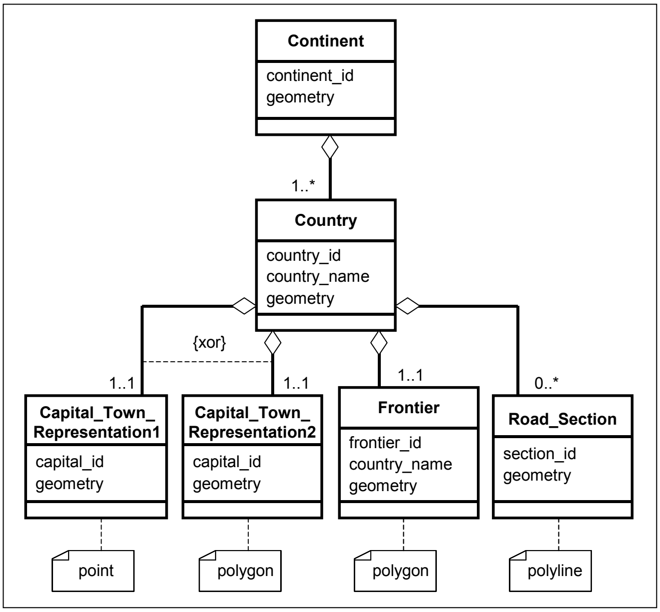


Figure 2: UML conceptual design of spatial aggregation.

In order to simplify the vocabulary of this paper, we define that whole tables contain whole instances and part tables contain part instances. The example of figure 3 illustrates the existing link between the geometry value of a whole instance (small_country) and the geometries of its part instances (capitalA, frontierB, sectionC, sectionD). Concerning the data storage, it appears that the whole geometry groups the part geometries in a single collection.

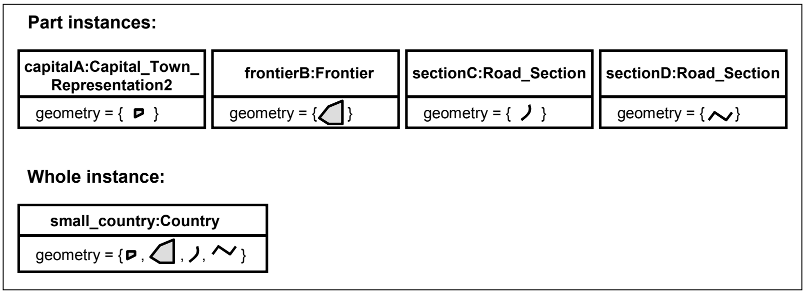


Figure 3: Exemple of aggregation: links between the whole instance geometry and its part instance geometries.

1.2 Problematics

Computer scientifists need to exactly know what types of geometries they will find in tables resulting from aggregation, particulary when they will write program code that queries the database. Indeed, they have to precisely identify what are the types of geometries (point, polyline,…) extracted by computer-based applications from whole instances. Thus, the main purpose of the works presented in this paper consists in defining solutions to deduce the geometry types stored in a whole from the geometry types of its parts. This computation is made from UML specifications of spatial aggregation operations in considering that geometry types of input tables are known. If we consider the example of figure 2, our objective is to propose a mechanism to determine spatial types allowed in the attribute geometry values of Country and Continent whole instances. In fact, our works respond to the question "what can contain values of the geometry attribute in a whole table ? one point ? several points ? several points and several polygons ? several points or several polygons ? the same number of points as the number of polygons ? …". Figure 4 schematizes and summarizes the purpose of the present paper.

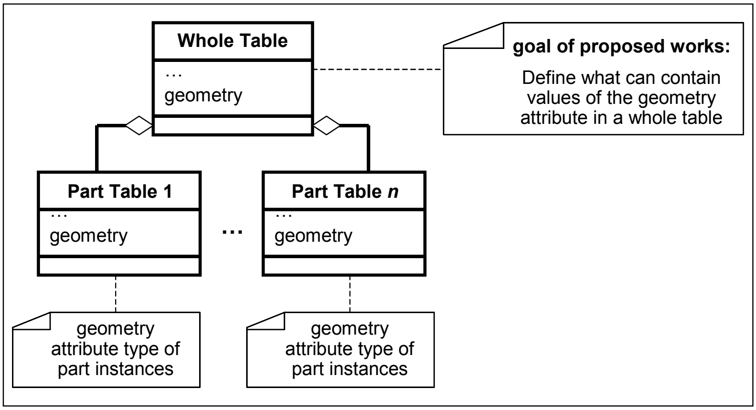


Figure 4: Goal of proposed works: determine spatial types of whole tables.

We propose two approaches to reach the presented goal. The first one, described in section 2, is theoretical and is based on a value domain study. In fact, we will formalize the link existing between the value domain associated to the geometry attribute of the whole table, and the value domains associated to the geometry attribute of the part tables. The second approach, described in section 3, is a practical method based on a numeric constraint representation of spatial types allowed in tables.

2. LINK BETWEEN A WHOLE AND ITS PARTS: APPROACH BASED ON VALUE DOMAINS

2.1 Notation

The presented formalism uses two types of unordered collections: the set and the bag. The set doesn't contain duplicate elements; any element can be represented only once. A set is denoted by $\{element_1, \ldots, element_n\}$. Let S be a set, 2^S is the set of all subsets of S. A bag is similar to a set, but it can contain duplicate elements; that is, the same element can occur in a bag more than once. A bag is denoted by $Bag\{element_1, \ldots, element_n\}$. The empty bag is $Bag\{\}$. The combination of bags is allowed by using the union operator. For example, $Bag\{a,b\} \cup Bag\{b,c\} \cup Bag\{\} = Bag\{a,b,b,c\}$. The cross-product (denoted by $\times$) provides the capability for combining sets of bags. Let S_1 and S_2 be two sets of bags, $S_1 \times S_2 = \{ Bag_i \cup Bag_j \mid Bag_i \in S_1 \text{ and } Bag_j \in S_2 \}$. For example, $\{ Bag\{a,b\}, Bag\{c\} \} \times \{ Bag\{c\} \} = \{ Bag\{a,b,c\}, Bag\{c,c\} \}$. The cross-product is associative. Also, because the bag is an unordered collection, the cross-product on sets of bags is commutative. S^n is the cross-product of S with itself n times; if applied to cross-product on a set of bags, $S^0 = \{Bag\{\}\}$; $S^1 = S$; $S^2 = S \times S$; $S^3 = S \times S \times S$;... A multiplicity is denoted by $(min..max)$ with $min, max \in [0..+\infty[$ and $min \leq max$. Also, $+\infty$ is denoted by *.

2.2 Value domains of geometry attributes

A geometry attribute can be characterized by its value domain i.e. the set of all possible values that a geometry attribute can have for a specific table. In this section, we will propose to bring out the link between value domains of part geometry attributes and the value domain of the whole geometry attribute. The final goal is to know the exact value domain of the whole geometry attribute, which gives useful information concerning spatial types allowed in the whole table.

A value domain of a geometry attribute is a set of bags and consequently, the value of a geometry attribute is a bag: generally, the main goal of geographic data is to be displayed on a map, and thus an order on geographic elements is not considered by the database designer.

Let *Dom*(t) be the domain of values related to a type t; if t $\in$ {point, polyline, polygon} then t is called a simple type. *Dom*(point), *Dom*(polyline) and *Dom*(polygon) are respectively the sets of points, polylines and polygons. For example, *Dom*(polygon) is the infinite set of all bags that contain only one polygon. More precisely, we define that:

$$Dom(\text{point}) = \{ Bag\{point_1\}, Bag\{point_2\}, \ldots, Bag\{point_i\}, \ldots \}$$
$$Dom(\text{polyline}) = \{ Bag\{polyline_1\}, Bag\{polyline_2\}, \ldots, Bag\{polyline_i\}, \ldots \}$$
$$Dom(\text{polygon}) = \{ Bag\{polygon_1\}, Bag\{polygon_2\}, \ldots, Bag\{polygon_i\}, \ldots \}$$

In sections 2.3-2.5, we will define a function in order to determine the domain related to the geometry attribute of the whole class from domains of geometry attribute associated to the part classes.

2.3 Formal structure for aggregation relationships

The first step is to describe a formal structure for aggregation relationships and xor relations existing between them.

DEFINITION 1. *Let* $P(C') = \{ R_1 = (C_1, min_1, max_1), \ldots, R_n = (C_n, min_n, max_n) \}$. $P(C')$ *is the set of relationships aggregating a class* C'*. For each relationships* R_i*, the part class is* C_i *and* min_i, max_i *are the multiplicity implied in* R_i.

Let $X(C') = \{ \{R_i, R_j\}, \ldots, \{R_u, R_v\} \}$. $X(C')$ *is the set of all binary xor relations between distinct elements of* $P(C')$.

Let $Dom_{Class}(C)$ *is the value domain associated to the geometry attribute of the class C.*

Because binary xor relations are symmetric and irreflexive, elements of $X(C')$ are sets having a size equal to 2. if we consider the Country class of figure 2,

$P(Country) = \{ R_1 = (Capital\_Town\_Representation1, 1, 1),$
$R_2 = (Capital\_Town\_Representation2, 1, 1),$
$R_3 = (Frontier, 1, 1), R_4 = (Road\_Section, 0, +\infty) \}$

$X(Country) = \{ \{R_1, R_2\} \}$

$Dom_{Class}(Capital\_Town\_Representation1) = Dom(\text{point})$

$Dom_{Class}(Capital\_Town\_Representation2) = Dom(\text{polygon})$

$Dom_{Class}(Frontier) = Dom(\text{polygon})$

$Dom_{Class}(Road\_Section) = Dom(\text{polyline})$

2.4 Multiplicity effect

The second step is to study the effect of the multiplicity on a value domain. This is given by the function Dom_{Mult} . As exemplifed in figure 5, the goal is to determine the value domain associated to the geometry attribute of a whole class from the value domain associated to the geometry attribute of a part class, in considering a multiplicity (*min..max*).

DEFINITION 2. *Let Dom_{Mult} be a function that applies a multiplicity (min..max) on a value domain. The result is a value domain.*

Let $R = (C, min, max)$ be an element of $P(C')$ and $d = Dom_{Class}(C)$,

$$Dom_{Mult}(R) = Dom(d)^{min} \cup \ldots \cup Dom(d)^{k} \cup \ldots \cup Dom(d)^{max} \text{ with } min \leq k \leq max$$

For $R_4 \in P(Country)$,

$$
\begin{aligned}
Dom_{Mult}(R_4) = {} & \\
& Dom(\text{polyline})^0 \cup Dom(\text{polyline})^1 \cup Dom(\text{polyline})^2 \cup Dom(\text{polyline})^3 \cup \ldots \\
= {} & \{Bag\{\}\} \cup \\
& Dom(\text{polyline}) \cup \\
& (Dom(\text{polyline}) \times Dom(\text{polyline})) \cup \\
& (Dom(\text{polyline}) \times Dom(\text{polyline}) \times Dom(\text{polyline})) \cup \ldots \\
= {} & \{Bag\{\}\} \cup \\
& \{Bag\{ polyline_1 \}, \ldots , Bag\{ polyline_i \} , \ldots\} \cup \\
& \{Bag\{ polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_i , polyline_j\} , \ldots\} \cup \\
& \{Bag\{ polyline_1 , polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_i , polyline_j , polyline_k \} , \ldots\} \cup \ldots \\
= {} & \{Bag\{\}, \\
& Bag\{ polyline_1 \}, \ldots , Bag\{ polyline_i \} , \ldots , \\
& Bag\{ polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_i , polyline_j\} , \ldots, \\
& Bag\{ polyline_1 , polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_i , polyline_j , polyline_k \} , \ldots\}
\end{aligned}
$$

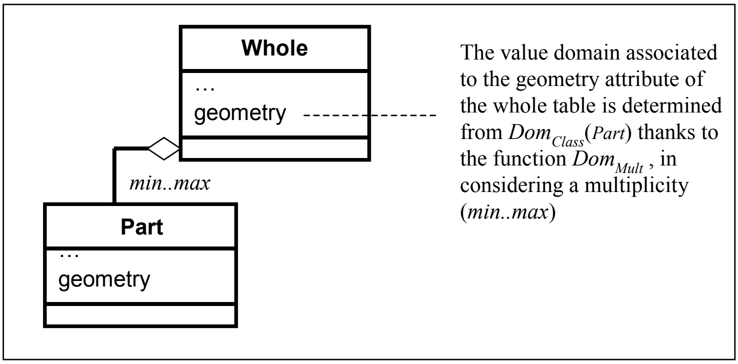


Figure 5: Dom_{Mult}, a function to determine the multiplicity effect.

For example, the application of the multiplicity (0..*) on *Dom*(polyline) (which is the part value domain) returns the infinite set of all bags that contain only polylines (which is the whole value domain) i.e. the infinite set of bags that contain from 0 to * polylines.

2.5 Link between the whole and its parts

The last step is to precisely define the link between the value domain of the whole and the value domain of its parts.

DEFINITION 3. *Let $Q_{\neg xor}(C')$ and $Q_{xor}(C')$ be two sets of sets. Each element of $Q_{\neg xor}(C')$ is composed of an relationship not implied in a xor relation.*
$Q_{\neg xor}(C') = \{ \{R_i\} \mid R_i \in P(C') \text{ and } \forall S \in X(C'), R_i \notin S \}$
Each element that is included in $Q_{xor}(C')$ is composed of relationships implied in a xor *relation.*
$Q_{xor}(C') \subseteq 2^{P(C')}$. *Let* $S \in 2^{P(C')}$, $S \in Q_{xor}(C')$ *iff*
$(\forall R_i, R_j \in S \text{ such as } R_i \neq R_j, \{R_i, R_j\} \in X(C'))$ *and*
$(\forall R_i \in P(C') \text{ such as } R_i \notin S, \exists R_j \in S \text{ such as } \{R_i, R_j\} \notin X(C'))$ *and* $|S| \geq 2$
Let $Q(C') = Q_{\neg xor}(C') \cup Q_{xor}(C')$.

In other terms, for each R_i that is not implied in a xor relation, $Q_{\neg xor}$ is used to "put" $\{ R_i \}$ in a set. For example, if $P(C') = \{R_a, R_b, R_c\}$ and if none of R_i is included in $X(C')$ then $Q_{\neg xor} = \{ \{R_a\}, \{R_b\}, \{R_c\} \}$. Moreover, $Q_{xor}(C')$ assembles binary xor relations in order to compose sets of *n*-arity xor relations. For example, if $X(C') = \{\{R_a, R_b\}, \{R_a, R_c\}, \{R_b, R_c\}\}$ then $Q_{xor}(C') = \{\{R_a, R_b, R_c\}\}$.

In the previous example,

$Q_{\neg xor}(Country)=\{\{R_3\},\{R_4\}\}$; $Q_{xor}(Country) = \{\{R_1, R_2\}\}$;
$Q(Country) = \{\{R_3\}, \{R_4\}, \{R_1, R_2\}\}$.

Finally, the value domain of the geometry attribute associated to a whole class can be defined from its parts by the function Dom_{Whole}. This function is defined below and takes as parameter $Q(C')$.

DEFINITION 4. *Let* $Q(C') = \{\{R_{11}, \ldots, R_{1m}\}, \ldots, \{R_{np}, \ldots, R_{nq}\}\}$. $Dom_{Whole}(Q(C')) =$
$(Dom_{Mult}(R_{11}) \cup \ldots \cup Dom_{Mult}(R_{1m})) \times \ldots \times (Dom_{Mult}(R_{np}) \cup \ldots \cup Dom_{Mult}(R_{nq}))$

More intuitively, $Q(C')$ can be viewed as a conjunctive normal form of relationships. Indeed, conjunction operations link the elements of $Q(C')$ and disjunction operations link elements of each set that is included in $Q(C')$. While the cross-product corresponds to a conjunction between value domains, the union is a disjunction of value domains. Each set included in $Q(C')$ and having a size ≥2 describes a xor relation i.e. a disjunction. Therefore, unions are defined between elements of each set included in $Q(C')$ in order to determine $Dom_{Whole}(Q(C'))$.

In the previous example, the value related to the geometry attribute of a Country instance is composed of "a polygon issued from R_3 and polylines issued from R_4 and (a point issued from R_1 (exclusive) or a polygon issued from R_2)". Thus, the function Dom_{Whole} applies the multiplicity on the value domain associated to the geometry attribute of each part class by using the function Dom_{Mult}. For example,

$Dom_{Whole}(Q(Country)) = Dom_{Mult}(R_3) \times Dom_{Mult}(R_4) \times (Dom_{Mult}(R_1) \cup Dom_{Mult}(R_2)) =$
$Dom(\text{polygon}) \times Dom_{Mult}(R_4) \times (Dom(\text{point}) \cup Dom(\text{polygon}))$
$= \{Bag\{ polygon_1 \}, \ldots , Bag\{ polygon_i \} , \ldots\} \times$
$\{Bag\{\},$
$Bag\{ polyline_1 \}, \ldots , Bag\{ polyline_j \} , \ldots ,$
$Bag\{ polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_j , polyline_k\} , \ldots,$
$Bag\{ polyline_1 , polyline_1 , polyline_1 \}, \ldots , Bag\{ polyline_j , polyline_k , polyline_m \} , \ldots\} \times$
$(\{Bag\{ point_1 \}, \ldots , Bag\{ point_p \} , \ldots\} \cup \{Bag\{ polygon_1 \}, \ldots , Bag\{ polygon_q\} , \ldots\})$
$= \{ Bag\{ polygon_1 , polygon_1 , point_1 \}, \ldots , Bag\{ polygon_i , polygon_q , point_p \} , \ldots ,$
$Bag\{ polygon_1 , polygon_1 , point_1 , polyline_1 \}, \ldots ,$
$Bag\{ polygon_i , polygon_q , point_p , polyline_j \} , \ldots ,$
$Bag\{ polygon_1 , polygon_1 , point_1 , polyline_1 , polyline_1 \}, \ldots ,$
$Bag\{ polygon_i , polygon_q , point_p , polyline_j , polyline_k\} , \ldots,$
$Bag\{ polygon_i , polygon_1 , point_1 , polyline_1 , polyline_1 , polyline_1 \}, \ldots ,$
$Bag\{ polygon_i , polygon_q , point_p , polyline_j , polyline_k , polyline_m \} , \ldots\}$

The result of $Dom_{Mult}(R_4)$ was previously presented. This example shows that thanks to Dom_{Whole}, a value domain can be divided into cross-products and unions between value domains related to a simple type (point, polyline, polygon). In the same way, the value domain associated to the geometry of the Continent class can be determined from the previous result.

$Dom_{Whole}(Q(Continent)) =$
$(Dom(\text{polygon}) \times Dom_{Mult}(R_4) \times (Dom(\text{point}) \cup Dom(\text{polygon})))^1 \cup$
$(Dom(\text{polygon}) \times Dom_{Mult}(R_4) \times (Dom(\text{point}) \cup Dom(\text{polygon})))^2 \cup$
$(Dom(\text{polygon}) \times Dom_{Mult}(R_4) \times (Dom(\text{point}) \cup Dom(\text{polygon})))^3 \cup$
…

To summarise this section, $P(C')$ and $X(C')$ are the formal structures of an aggregation. C' is the whole class. Dom_{Whole} is a function that determines the value domain of the geometry associated to C'. Dom_{Whole} uses both the function Dom_{Mult} and the set $Q(C')$ constructed from $P(C')$ and $X(C')$.

3. LINK BETWEEN A WHOLE AND ITS PARTS: APPROACH BASED ON NUMERIC CONSTRAINTS

The previous section offers theoretical fundations concerning the link existing between a whole and its parts. Unfortunately, it is often impossible or not reasonable to compute directly cross-products and unions of value domains in order to deduce the whole; in other words, it is often impossible or not reasonable to use Dom_{Whole} to determine the whole. Thus, this subsection presents intuitively a finite form of value domains related to geometry attributes. This finite form is suitable for computing the whole from its parts. Indeed, as presented in figure 6, each value domain can be represented by a set of numeric constraints. We consider two part classes (P1 and P2) having a specific value domain; we suppose that these tables have been produced by other aggregation operations.

The first set of constraints associated to classes consists of inequalities that give absolute boundaries for each simple type. The general form of these inequalities is $min \leq n(t) \leq max$ where n(t) is number of elements having the simple type t in a bag of the considered value domain. *min* and *max* are the boundaries of n(t). For example, in numeric constraints of class P1 presented in figure 6, the inequalities show that each bag of the considered value domain contains from 0 to 8 polylines, and from 0 to 8 points. In the same way, the numeric constraints of class P2 in figure 6 show that each bag of the considered value domain contains (2a) from 1 to 10 polylines and from 1 to 10 polygons, or (2b) only from 1 to 10 polygons.

The second part of numeric constraints is composed of equalities that underline the dependencies existing between element numbers of each bag. For example, a dependency exists between the number of polylines and the number of points in each bag included in the value domain of the class P1 (figure 6). This is due to the fact that a multiplicity is applied on a conjunction between a type polyline and a type point. This leads to an equality between the number of polylines and the number of points. In other words, the number of polylines and the number of points are the same in each bag of the considered value domain.

The whole class W in figure 6 corresponds to the aggregation of the class P1 and the class P2. The constraints of the class W consist of the union between the constraints of the class P1 and the constraints of the class P2. It is important to notice that this method cannot be directly applied in the case of dependent simple types i.e. if two distinct types are dependent. Two simple types t_1 and t_2 are dependent iff $Dom(t_1) \cap Dom(t_2) \neq \emptyset$.

4. CONCLUSIONS AND PERSPECTIVES

The contribution of this paper is to provide a method for the computation of spatial types resulting from aggregation operations. Indeed, database users need to exactly know what types of geometries are stored in tables resulting from spatial aggregation. Thus, we propose both a complete theoretical formalization and a practical study concerning the deduction of the spatial type associated to a whole table. The computation is made from the UML conceptual specifications of aggregation operations generally used in the field of GIS (Bédard, 1999; Borges, 1999; Brodeur, 2000; Jugurta, 1999; Lbath, 2000; Laurini, 2001).

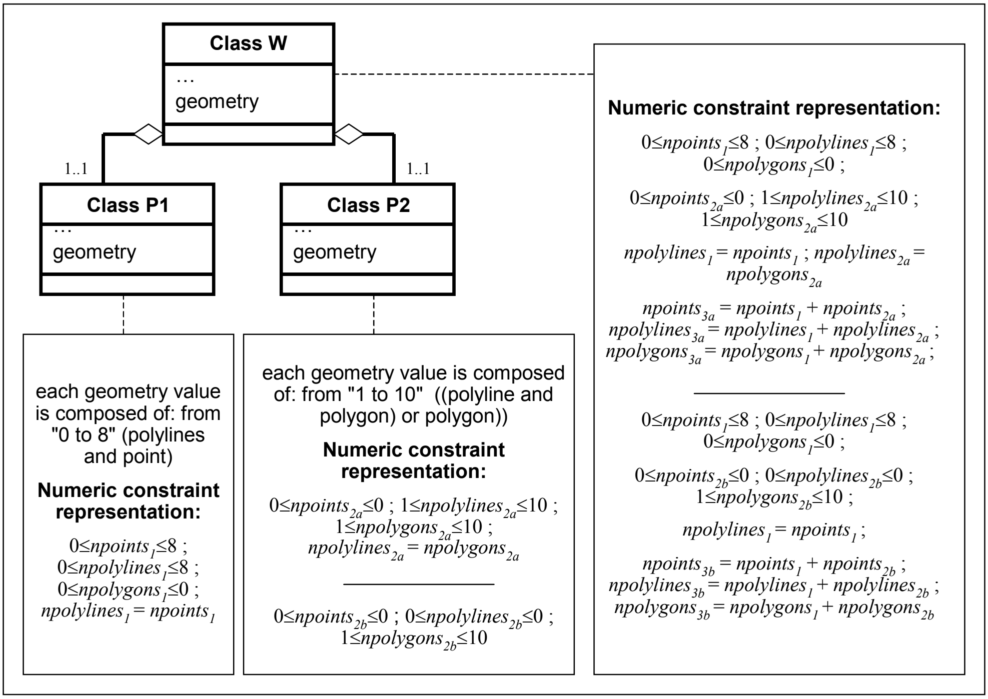


Figure 6: Representation of value domains by numeric constraints.

Indeed, firstly, we defined a formalization to describe the theoretical link between the whole value domain and the part value domains. Secondly, we proposed an operational method based on numeric constraints in order to represent and compute aggregations. The main goal of this paper is to lead to the proposition of an efficient algorithm for the automatic generation of whole types resulting from aggregations. We underline, in this paper, an application that concerns the spatial aggregation but other types of aggregations can be considered in various fields. Simple types used in this paper (point, polyline, polygon) can be changed in order to respond to other designer requirements.

We plan to apply the work presented here, to the step of the query results grouping in a data integration system. A new generation of data integration systems (Peer Data Management System - PDMS) allows the cooperation of data sources in an extensible architecture. It is the increase of the P2P network that inspired such systems; a P2P system is a virtual network which allows for resources to be shared between autonomous systems without using a global schema. At Cemagref and LIMOS, we work on a PDMS based on a SuperPeer approach (Jaudoin, 2004). The different data sources are gathered in clusters (called communities) around a SuperPeer; the SuperPeer maintains the community knowledge and is responsible for the query resolution (in our case it is a problem of query rewriting in terms of views). The data sources can cooperate in one community via the common knowledge of the community and between communities thanks import of knowledge part from others communities.

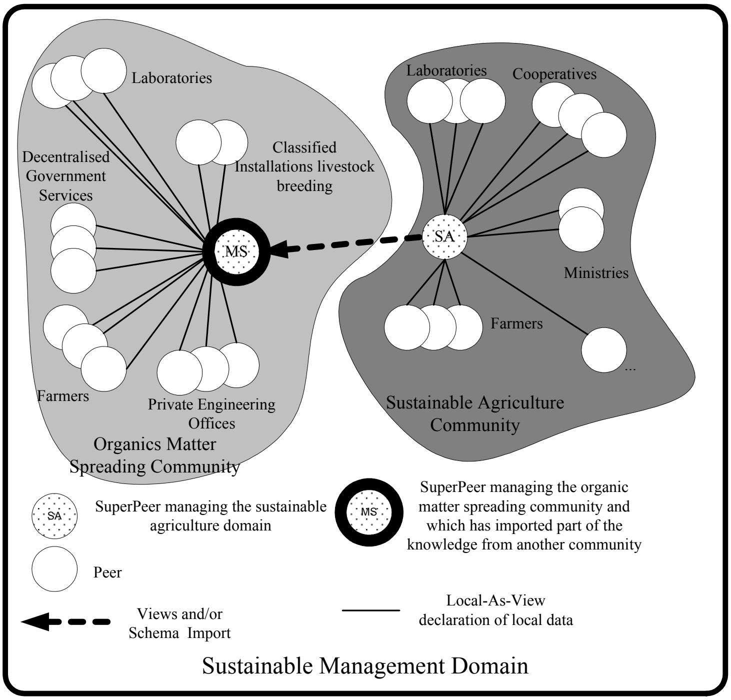


Figure 7: General domain concerning sustainable management compound of communities.

Figure 7 shows the general domain concerning sustainable management compound of communities. A query can be solved by the data sources of one community; it can also require supplementary communities and thus need the querying of other data sources from these supplementary communities. The works presented here is very interesting for the grouping of the query results distributed through the data sources. Indeed the query is likely to require many data sources to achieve its resolution and each of data sources returns a part of the global answer. The different parts have to be gathered and "composed" in order to form the global answer.

In parallel of the presented works, a user-oriented representation of geographic type specifications in class diagrams is being investigated and a first version of a UML profile for geographic databases has been defined. In this UML extension, for each class that has a geographic feature, spatial types are declared thanks to UML tagged values and stereotypes.

BIBLIOGRAPHY

Bédard Y., Visual Modeling of Spatial Databases Towards Spatial Extensions and UML. Geomatica: Vol 53(2). 169-186, 1999.

Booch G., Rumbaugh J., Jacobson I., The Unified Modeling Language User Guide. Addison-Wesley: 482p, 1998.

Borges K., Laender A., Clodoveu D., Spatial Data Integrity Constraints in Object Oriented Geographic Data Modeling. In Proc. of the Int. ACM Symposium on Advances in Geographic Information System. USA, 1-6, 1999.

Brodeur J., Bédard Y., Proulx M.J., Modelling Geospatial Application Databases using UML-based Repositories Aligned with International Standards in Geomatics. In Proc. of the Int. ACM Symposium on Advances in Geographic Information Systems. USA, 39-46, 2000.

Jaudoin H., Schneider M., F. Vigier, Querying Data Sources in a SuperPeer Data Management System. In Proc. of Int. Conference On Flexible Query Answering Systems. France, to appear, 2004.

Jugurta L.F., Cirano L., Specifying Analysis Patterns for Geographic Databases on the Basis of a Conceptual Framework. In Proc. of the Int. ACM Symposium on Advances in Geographic Information System. USA, 7-13, 1999.

Kösters G., Pagel B., Six H., GIS-Application Development with GeoOOA. Int. Journal of Geographical Information Science: Vol.11(4). 307-335, 1997.

Laurini, R., Information Systems for Urban Planning. Taylor & Francis: 308p, 2001.

Lbath A., Pinet F., Towards Conceptual Modelling of TeleGeoProcessing Applications. In Proc. of the Int. Symposium on TeleGeoProcessing. France, 25-39, 2000.

MapInfo Corporation, MapInfo Professional user's guide. USA, 2003.

Tryfona N., Pfoser D., Hadzilacos T.: Modeling Behavior of Geographic Objects: An Experience with Object Modeling Technique. In Proc. of the Int. Conference on Advanced Information Systems Engineering. Spain, 347-359, 1997