
The stick operator: the management of several points of view in a geographical information system

Géraldine Del Mondo* and
Michel Mainguenaud

INSA Rouen Normandie Univ.,
UNIROUEN, UNIHAVRE,
LITIS, 76000 Rouen, France
Email: geraldine.del.mondo@insa-rouen.fr
Email: michel.mainguenaud@insa-rouen.fr

*Corresponding author

Abstract: In this article, we propose the *Stick* operator which provides a unique point of view from two points of view in the context of a geographical information system. It is based on two alphanumeric data models we merge in order to obtain a single data model (i.e., the result of the *Stick* operator). This operator uses a set of rules defined on characteristics of data models associated with each point of view. These characteristics are based on the links between the spatial dimension and alphanumeric data. It can be used when two different levels of details/abstraction or two sources have to be merged. We rely on a labelled graph to formalise data models and work till the attribute level. The *Stick* operator takes advantage of characteristics such as the intersection of graphs, the existence of quasi-strongly connected graphs or paths to provide an extended result data model.

Keywords: semantic alphanumeric data; levels of abstraction; geographical information systems; data model merge; graph data model; database view; database snapshot.

Reference to this paper should be made as follows: Del Mondo, G. and Mainguenaud, M. (2019) 'The stick operator: the management of several points of view in a geographical information system', *Int. J. Spatial, Temporal and Multimedia Information Systems*, Vol. 1, No. 3, pp.197–231.

Biographical notes: Géraldine Del Mondo received her PhD in Geomatic/Computer Science from the University of Brittany in 2011, after three years working at the Naval Academy Research Institute in Brest, France. She is currently an Assistant Professor at the Department of Information System Engineering of the National Institute of Applied Science (INSA) at the University of Normandy. She is a member of the Transport Intelligent System group (STI) of the Computer Science, Information Processing, and Systems Laboratory (LITIS). Her research interests include geomatic, spatial and temporal reasoning, databases, applied graph theory and intelligent transport systems.

Michel Mainguenaud was the Director of the Information System Engineering Department and then the Dean for Education at the Institut National des Sciences Appliquées (INSA) of Rouen. He was the Director of the National CNRS/SIGMA-Cassini research group (a CNRS recognised national group to promote research in geographical information systems in France). His main interests are geographical information systems, multi-media database models and query languages: documents, images and maps.

1 Introduction

The growing interest for big data reinforces researches in geographic information sciences. Indeed, numerous manipulated data have a (geo-)localisation. Data manipulations and analysis of geographic data are becoming more and more meaningful. As examples, we can provide: volunteered geographic information (VGI) and uploads of cooperative data from smartphones or embedded GPS with location-based applications. The abstraction levels are very variable: from the world, countries, regions, ... to individuals. Within all these levels, more and more applications use multi-sources data. Therefore, these data have to be linked in order to provide a more accurate analysis or a better fitted service. The definition of operators with a multi-level of abstraction approach (from the semantic or the spatial points of view) is still an open problem. Defining closed operators (in order to allow composition of operators) provides the opportunity to design high level operators. They can be expressed as a composition of lower level operators. Therefore it appears two main approaches. The first one is a bottom-up approach, starting from the cartographic elements to the data model abstraction. The second one is a top-down approach, starting from data models to (possibly) join cartographic elements. This article is oriented toward a top-down approach.

As an example and without loss of generality, we illustrate this problem with a navigation aid tool. Like embedded software's in vehicles, known by abuse of language embedded GPS, the management of areas with navigation difficulties is subject to special treatments. On arrival near an area where the number of interchanges, streets or other drop-off areas is important, mapping (although minimal) is improved in order to allow a better visibility of the environment. Drivers then have a representation with a greater level of detail to guide them in their decisions. Once the area where risks of error are important, is behind the car, the display is presented as usual with a greater spatial coverage and a lower level of detail.

However, for a movement with a high relative speed (e.g., vehicles, high-speed trains) or a slow one (e.g., pedestrians), the problem of providing semantic information, and not only a map [as in Mustière and Devogele (2008) remains open Olteanu-Raimond and Mustière (2008)]. Many tools are now available, both in vehicles and on mobile phones for example. They are essentially based on a cartographic approach for the representation of space. The semantic part is relatively absent from these proposals. Two treatment levels can be highlighted. A first level concerns the mapping of the space and its treatments (e.g., generalisation, graphic semiotics, geographic name placements). A second level concerns the data model to manage semantic information (i.e., alphanumeric data) associated with objects in this space. Undoubtedly, this work is positioned at the second level.

On-board navigation tools provide alphanumeric information from a single source. The development of interoperability leads to the composition of different models. The composition of these models provides enhanced information to end-users. Thus, the addition of semantics may come from different applications working at the same level of abstraction in the context of an interoperable process. This is the case with the development of servers based on Web Map Service/Web Feature Service (WMS/WFS) specifications. Their main aim is to provide free or paid cartographic representations and associated alphanumeric information. The addition of semantic may also result from a change of level of granularity during data manipulations. Approaches such as those of

Stell and Worboys (1998) or Aiken et al. (1996) are an illustration. Semantic additions may also come from a mixed approach as a combination of several perspectives into a common vision. This principle is similar to the process of view integration in the context of relational databases (Ullman, 1988). It can be developed as a vision for interoperability (i.e., exogenous schemas to be shared to offer a vision of a geographic location) or by bringing the perspective of a single information system (i.e., endogenous schemas to be shared to provide additional information). In both cases, we face the need for an operator to glue these data models.

In this article, we propose to tackle the management of alphanumeric information provided to an end-user with the *Stick* operator. This operator aims to enable a merge between different perceptions of an area in terms of semantic information associated with a spatial representation. It has to be complemented by an operator handling map information, for the visual representation of the considered space (e.g., displaying map objects obtained by various generalisations, changing the legend). Only the *Stick* operator is defined in this article.

In order to take into account a larger and larger volume of data (which may not be structured), two main approaches of architecture are designed: the mediation or the federation. The used architecture is orthogonal to the approach we present in this article. Therefore, we do not pay attention to this architecture. For the formalisation of a data model, two main approaches are available: the management of ontologies on which data models are designed (Cruz and Xiao, 2008) and the artefacts for integration of models (Aiken et al., 1996; Estublier et al., 2008). Ontologies and the semantic management rely on information linked to the application or the geometry (e.g., below, above) but at the class, entity level (whatever the name provided by the model is). We assume here that the first point is resolved using aligned ontologies. Similarly to the relational model and its view integration process for conception, this operation may be used for a wide spectrum from the step by step integration (i.e., view by view) to the whole integration using a single step. So data models, eventually provided by different sources, do not present ambiguities. Artefacts of modelling propose a framework to solve this problem, on which we rely on. The main difference with other approaches in the literature is based on the fact that we manage the process in this article at the attribute level.

The *Stick* operator is designed in the following context: as a binary operator and at the attribute level. We can then focus on the integration of artefacts to provide as many relevant information as possible to an end-user based on available data within the (two) data models. These models can be independent or may have common concepts.

We use a data model taking into account the spatial relevance of information with respect to the logic spatial representation (and not from the cartographic point of view). An orthogonal research area to a spatial perception is the introduction of the temporal component in the navigation aid process for example. Although this component is important, our proposal is oriented towards the provided information at a given time t on the way. Only the spatial component is used in this article.

We define a set of rules to build the data model associated with the result of the *Stick* operator. We rely on: a class model represented by a labelled graph with properties such as inheritance, (strict) spatial inclusion, cardinality constraints and links between the spatial dimension and alphanumeric data. From a reference graph, the result of the *Stick* operator between two sub-graphs is still a graph.

Nodes and arcs are defined taking into account properties such as the empty intersection (or not) of the two initial sub-graphs, the existence of a string/path (or not) between two sub-graphs in the reference graph.

Section 2 presents a state of the art of approaches for data model integrations linked with our issues. Section 3 presents the principles of the *Stick* operator. Section 4 presents the treatment of independent data models. Section 5 presents the treatment of non-independent data models. Section 6 presents a conclusion of our works and their perspectives.

2 State of the art

Our problem deals with (alphanumeric) data models managing information in the context of a spatial environment. About interoperability, many works associated with (or not) the Open Geospatial Consortium have the objectives to allow exchanges of data/processing between different agencies (Lieberman et al., 2007; OGC-WM, 2006). One of the main directions is based on ontology alignments (Pavel and Euzenat, 2013; Partyka et al., 2008). An ontology alignment covers two aspects: The discovery of correspondences between different ontologies (e.g., equivalence of concepts and relationships, subsumption) and the expression of these connections through a data model. Semantic interpretation problems are located at this second aspect (Harel and Rumpe, 2004). In this work, we consider that this processing step is already performed with a coherent alignment of models.

The model driven engineering (MDE) is a field of computer sciences and provides tools, concepts and languages to create and to transform models. The additions of information from different sources or from the same source but with different perspectives (i.e., different levels of granularity) are merged into a single scientific problem: the composition of models.

Firstly, we study proposed solutions with a formal modelling approach, to this issue, secondly, the operational implementation of these solutions, and we end with a synthesis of these solutions.

2.1 Formal approach

The formal approach relies on the definition of an algebraic model, the management of different levels of granularity and the integration of artefacts.

2.1.1 Algebraic model

Herrmann et al. (2007) propose an algebraic approach for the composition of models. In its simplest way, the composition of models is a mechanism that takes as an input, two models and provides as a result a third one. The starting point is a link between a model and an application domain. This approach defines a function (sm) which is applied on a modelling language (M) and provides a semantic domain (D) such as for every model, $m \in M$, it must be in accordance with a domain D:

$$\text{sm} : M \rightarrow D$$

As an example, the modelling language may be UML and the semantic domain may be a navigation aid application. The language allows modelling components such as towns and communication links.

The starting hypothesis is the consistency of models regarding applications and a model m_2 refines a second one m_1 if:

$$sm(m_2) \subseteq sm(m_1)$$

The interpretation is that the number of systems implementing the model m_2 is lesser than the number of systems implementing the model m_1 .

They define a binary operator (\otimes) which, applied to two models, produces a composed model:

$$\otimes : M \times M \rightarrow M$$

With this operator, they define three properties about the preservation of coherency: ‘property preserving’ (PP), ‘full property preserving’ (FPP) and ‘consistency preserving’ (CP).

The first property (PP) relies on:

$$PP : \forall m_1, m_2 \in M : sm(m_1 \otimes m_2) \subseteq sm(m_1) \wedge sm(m_1 \otimes m_2) \subseteq sm(m_2)$$

This property guarantees that no information available in the first (resp. the second) model is lost during the composition.

The notion of \subseteq is related to the fact that information provided by the composition (i.e., \otimes) can be added although that information did not belong to any model before the composition (m_1 neither m_2).

The second property (FPP) relies on:

$$FPP : \forall m_1, m_2 \in M : sm(m_1 \otimes m_2) = sm(m_1) \cap sm(m_2)$$

This property allows to analyse the result data model from each original model since the definition is built with the equality. The modifications associated with a model acting as an operand has an impact on the part of the result data model provided by this model.

The third property (CP) relies on:

$$CP : \forall m_1, m_2 \in M : sm(m_1) \cap sm(m_2) \neq \emptyset \Rightarrow sm(m_1 \otimes m_2) \neq \emptyset$$

The addition of information, due to the composition may render inconsistent the result data model (i.e., $sm(m_1 \otimes m_2) = \emptyset$) even if initially the models (each taken individually) were not inconsistent (i.e., $sm(m_1) \cap sm(m_2) = \emptyset$).

Using this formal definition of a data model composition, two main approaches may be defined to make this composition a reality: the structure by levels of granularity and the composition by the integration of artefacts.

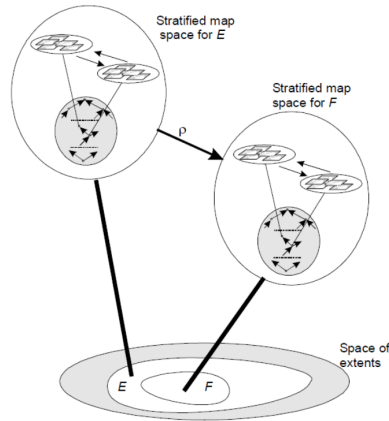
2.1.2 Levels of granularity

Approaches linked to the management of several levels of abstraction are mandatory for the global understanding of phenomena (Longo and Medeiros, 2013; Adnani et al., 2001). Works defined in Stell and Worboys (1998) follow this philosophy. They present the

advantage of providing a high level conceptual framework in order to manipulate several levels of granularity (i.e., m_i presented in Section 2.1.1). The authors define a function, ρ . This function manages passages from a level to another one (Figure 1). This framework does not require any specificity to model data as in Fonseca et al. (2002) and relies on the following concepts:

- *Map*: A data collection (with geometric and semantic attributes if it is possible). It is a database state. As an example, a paper map is a map as defined in their works.
- *Map space*: Set of *Map*. That means all instances of a database with a specific schema. They are partially ordered depending on the semantic or spatial precision (in the sense that all information from a fixed database schema may not be available for a given map). As an example, an instance of a CityGML schema is an interpretation of a reality and represents the map space. Each map, represented following this schema, will be an element of this map space.
- *Granularity lattice*: Allows to define an order over *Map* such as levels of granularity from the spatial and semantic points of view vary within a threshold, a minimum and a maximum (two different lattices). As an example, all information (provided from a fixed schema) will not be available for a given map (i.e., an instance of a CityGML schema). Therefore maps may be partially ordered depending on the level of information they offer.
- *Extent (E)* of a representation: the set of elements that belongs to this representation. The extent may be geometric or semantic oriented. As an example, it represents the set of element of the database schema which is available for this map (from the semantic point of view) and the set of element represented on the map (e.g., areas, points, and a legend).
- *Stratified map space (SMS) $\Sigma(E)$* : This concept allows to manage the different levels of definition (from the semantic or spatial points of view). A SMS is a *granularity lattice* such as elements are map spaces. The idea is to order, within a lattice, different map spaces depending on their (spatial and/or) semantic granularity. As an example, an OpenStreetMap will provide different levels of detail in the definition of manipulated data. The SMS is a lattice defined by the different schemas (i.e., for the different levels of detail). The variation within different granularity levels is possible using two operations defined between map spaces: the generalisation, *Gen*, and its opposite, *Lift*. A SMS is associated with a unique extent (i.e., the extent is unchanged).
- *Sheaf of stratified map space*: Allows to avoid being limited to a single extent and defines relationships between SMS. Each SMS is associated with an extent. A sheaf of stratified map space is a set of SMS. The ρ function allows the passage from one SMS to another one. As an example, the goal here would be to manipulate maps provided by different providers on the common space they describe.

The static aspect of the model is provided by the different structures (*extent*, *SMS*). The dynamic aspect of the model is provided by the ρ function. In this article we focus more precisely on the dynamic aspect from the semantic point of view [this level is not present in the Stell and Worboys (1998) proposition]. We define our work at the level of *sheaf of stratified map space* in order to link different *extent(s)*.

Figure 1 Sheaf of stratified map space from

Source: Stell and Worboys (1998)

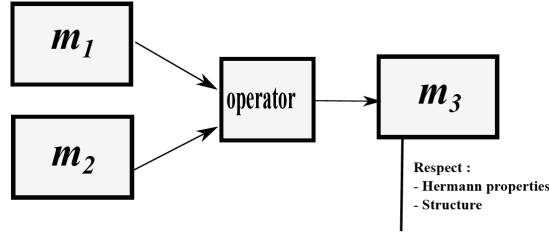
2.1.3 Integration of artefacts

The integration of artefacts (i.e., m_i defined in Section 2.1.1) is based on the management of heterogeneous models from the level of detail point of view (i.e., granularity level). The aim is to provide a homogeneous model called domain (Estublier et al., 2008).

The integration may have a top-down oriented approach or a bottom-up one. Mainly used in software engineering, it allows in a top-down orientation to identify programming artefacts without taking into account technologies that will be used (i.e., here, the equivalent of the public signatures and therefore the interface model). In a bottom-up approach, the designer must define the software artefacts and the consistency of their interactions. Whatever the orientation is, the synchronisation between software artefacts, and their models corresponds to actions performed by the software artefacts.

The work presented in Aiken et al. (1996) takes into account the semantic aspects. It proposes to end-users to create their own programs using a dedicated language in order to visualise data depending on the desired granularity level. The idea is to navigate into multi-dimensional data using the concept of view (different dimensions or levels of granularity) according to end-users' needs. Nevertheless, the design and the development of these programs, even if they use a graphical language is not so obvious. Actions to maintain the coherency between several levels of granularity is not clearly defined.

A mean to guaranty this coherency within a given context (i.e., Figure 2), is to propose operators that starting from potentially different models (e.g., m_1, m_2) are able to provide a result data model (e.g., m_3) that respects the formal properties defined in 2.1.1 and a coherent data structure (i.e., closed on the domain).

Figure 2 General skeleton

2.2 Operational management

The development of interoperability leads to the composition of different models provided by different sources. The composition of these models provides enhanced data to end-users. It appears therefore two main approaches to materialise a link between two levels of perception from the semantic information point of view. The first approach, named interpretation, relies on an ad-hoc construction of the space with a focus on a relevant interest point. The second approach, named compilation, relies on the global construction of possible interactions between two levels of space perception the compilation is the application for the whole set of interest points (i.e., the overlay area).

2.2.1 Interpretation

This approach is similar to the graphic management of software's dedicated to navigation aid in vehicles (but at the data model level instead of being at the spatial representation level). For a given data model and a given spatial place (i.e., an instance of a data model, a *Map* in 2.1.2), the data model associated with this instance is enhanced with an external input. It is the merge of the associated data model and instances which are concerned.

Since the interpretation approach uses a specific point of view, the public signature of (this 'merge' of data models) operation, named *Stick*, is:

$$\text{Stick} : \text{Model}_1 \times \text{Model}_2 \times \text{Interest Point} \rightarrow \text{Result Model}$$

This focus avoids a complete re-computation. The validity associated with the result data model is therefore relevant for a specific interest point. As an example, let us imagine two highways to enter a town A, defined on two interest points (spatially different but identical from the semantic point of view) B and C. This may be the case for an arrival using a highway in the North of a town and in the South of the same town (A). The *Stick* operator may not provide the same data model associated with B and C (even if it is based on the same instance A).

An alternative to this approach is to propose a more global vision of the *Stick* operator, using a generalisation of interest points, to the set of possible interactions between two data models.

2.2.2 Compilation

The approach, named compilation, merges two data models whatever the number of interest points is. The proposition of Stell and Worboys (1998) follows this approach. Two levels may be defined for the model manipulation. The first level is the notion of interoperability. It takes several models and tries to merge them. The second level uses external schemas, defined on the same reference model, which are the basic elements of this merging.

The first level raises the more general problem of multi-sources and the ontological alignment which is associated with. Numerous works tackled this question (Cruz and Xiao, 2008; Zhao et al., 2008). These works deal with, for example, homonym, synonym, a concept overlaying a concept in a different ontology... Cruz et al. (2004) proposes a solution based on a semi-automatic mapping providing a quality indicator for this alignment (e.g., exact, approaching, no link). Let us suppose that in the first level, the alignment of ontologies has already been performed (sine qua none condition for a 'merge'), we can focus on data model manipulations in a unified framework. The second level does not face the same problem since the reference model is the same one. Nevertheless, as in the first level, redundancy may appear since external schemas have no reason to be defined on exactly the same concepts.

The compilation approach generalises the public signature of the *Stick* operator defined in the interpretation approach:

$$Stick : Model_1 \times Model_2 \rightarrow \text{Result Model}$$

2.3 Synthesis

The algebraic model proposed in Section 2.1.1 allows to define properties a data model must verify. A result data model is built by adding or not information while using spatial operators. The approach defined in Stell and Worboys (1998) relies on several granularity levels but do not dynamically transfer information from one level to another one since it is entirely based on the compilation approach.

We take advantage of the Herrmann et al. (2007) properties, the Stell's model and we use an interpreted approach (i.e., the evaluation of a result data model for each demand). Therefore, it is possible to define operators that may be used with a philosophy close to the Aiden's one (i.e., by integration of artefacts).

The *Stick* operator manages attributes whenever a variation of abstraction levels is required. Aiken et al. (1996) let the responsibility of the result data model coherency to end-users. The definition of coherency rules for a transfer of information from a level of abstraction to another one provides a real added value as soon as its application is (semi-)automatic. Taking advantage of the notion of graph in order to be homogeneous with the *ZoomIn* operator (Del Mondo and Mainguenaud, 2016), and since graph data models are very interesting in this context (Hoel et al., 2015), the *Stick* operator can be formalised as a data model merge.

3 Principles of the *Stick* operator

The main goal of the *Stick* operator is to provide from two data models, a third one that takes advantage of links between the spatial components and the semantic components.

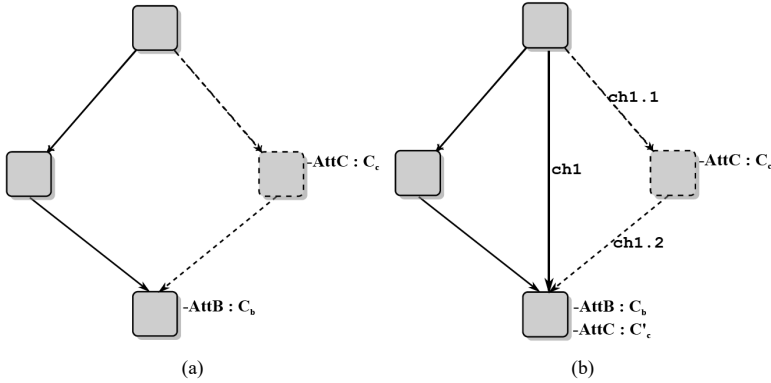
In this part we present the main elements of the *Stick* operator by defining the application context with a toy reference data model on which we rely to illustrate our examples, by defining the principles of the *Stick* operator and the presentation of the formalism we use.

3.1 Context

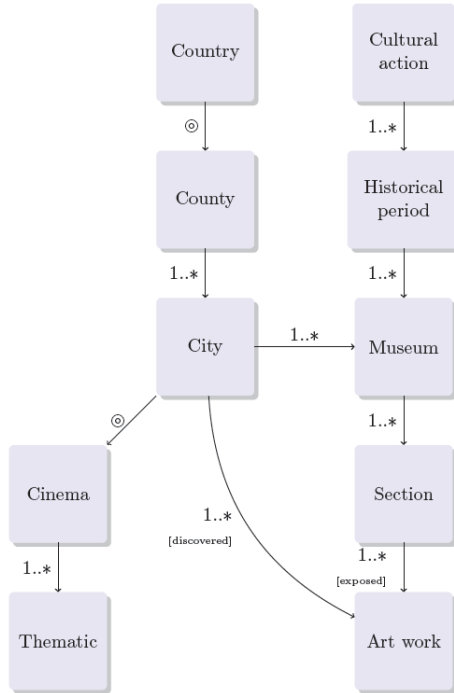
Let m be a reference data model (which may be obtained by different data models on which an ontologic alignment has been performed). We use the graph concept to formalise this data model. A directed graph is conventionally defined as $\mathcal{G} = (N, E, \psi, \nu, \epsilon)$ with N the set of nodes, E the set of arcs (i.e., $N \times N$), ψ the incidence function that associates nodes with arcs (i.e., multi-graph), ν (resp. ϵ) the labelling function for nodes (resp. arcs), managed as a set.

Let g be a sub-graph of \mathcal{G} (i.e., a view of m or a model m_i of Section 2.1.1), without circuit in which nodes and arcs may have been removed from the initial graph [e.g., Figure 3(a)]. In this case, there exists a loss of information (in term of attributes) between \mathcal{G} and g .

Figure 3 Deletion of a node in \mathcal{G}



In order to solve this problem, we proposed (Del Mondo and Mainguenaud, 2016) a step based on a propagation of attributes before the creation of a view [Figure 3(b)]. This step follows a set of rules, \mathbb{R}_p . These rules take into account the relationships between nodes (i.e., the label of a path, ch , between two nodes of the reference model graph). The labelling is based on a regular expression associated with a path. These rules allow to decide whether an attribute has to be propagated or not from a node to another one and eventually to modify its characteristic (GLOBAL, SUBSET, FUZZY) if needed.

Figure 4 Reference data model (see online version for colours)

For memory, an attribute is characterised as GLOBAL if its semantic validity is relevant for the global associated spatial representation (e.g., a population defined as an extension attribute for a town). An attribute is characterised as SUBSET if its semantic validity is relevant for any part of the associated spatial representation (e.g., the mayor's name of a town). An attribute is characterised as FUZZY if its semantic validity can not be insured. Its semantic validity is associated with a belonging function, \mathcal{A} , defined in $[0, 1]$.

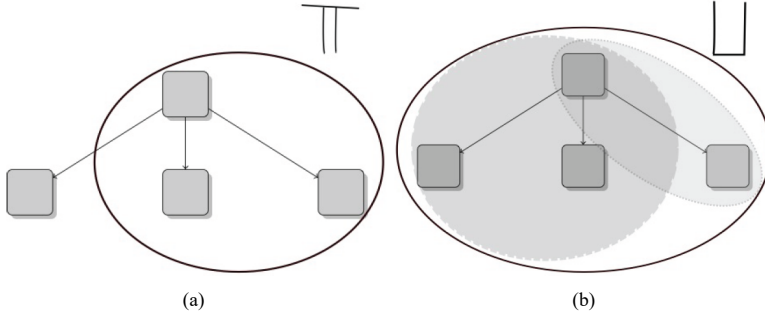
In Figure 3(a) without propagation, the attribute **AttC** characterised by C_c is lost. Propagation rules allows to decide whether or not an attribute should be propagated from one node to another, and to modify or not its characteristic (i.e., GLOBAL, SUBSET, FUZZY). In Figure 3(b) with propagation, **AttC** is kept (in respect with rules \mathbb{R}_p). This propagation may change its characteristic (C'_c). The propagation not only allows a transfer of attributes, but it also allows, due to the modification of their characteristic, the semantic coherency for nodes in which attributes are propagated. Once this step has been realised, the application of operators (e.g., *ZoomIn*, *Stick*) is now possible.

We use as an applicative domain, in order to illustrate examples, the administrative management of cultural resources. The reference data model is presented in Figure 4. Roles of relationships, whenever they are used, are detailed on arcs as [role]. The

labelling function associated with arcs (ϵ) models three relationships (used in \mathbb{R}_p): inheritance, (strict) spatial inclusion, and cardinality. These three relationships are respectively denoted using H , \odot and $1..*$.

In previous works, we defined a *ZoomIn* operator which tends to represent the projection (Π) operator of the relational algebra [Figure 5(a)]. As a result of its application to a graph g_1 , a sub-graph g_2 respecting the structure (i.e., classes) of a chosen projection is obtained. Thanks to the propagation step, g_2 leads to a maximum of information (including those that belonged to deleted nodes which have been removed by the application of the *ZoomIn* operator). We now present a new operator, the *Stick* operator which tends to ‘merge’ two data models [Figure 5(b)].

Figure 5 (a) *ZoomIn* (projection) (b) *Stick* (union)



3.2 The *Stick* operator

The *Stick* operator is from the semantic point of view close to the Union operator (\cup) applied to two graphs [Figure 5(b)]. We work at the data model level. This operator is defined as: binary, symmetric, closed (in order to allow the composition of operators) and set-oriented. It allows to improve the amount of information provided in a data model to facilitate a spatial analysis of a phenomenon.

It exists two main approaches for the specification of this operator. The first one is its application in the context of a unique reference graph. The second one is its application in the context of interoperability using two different reference graphs (but which have been previously aligned). In fact, these two cases can be generalised.

3.2.1 A unique reference data model

Within this approach, g_1 and g_2 are built from a unique graph \mathcal{G} . The application of the *Stick* operator is similar to look for at least one path in \mathcal{G} between a node of g_1 and a node of g_2 .

The labelling of this path is formalised with a regular expression built from labels of arcs defining the path. Two kinds of cases are to be studied: linked to the position (i.e., depends on the relationships) and linked to the semantics (depends on the choice of

attributes and their characteristics). The graph \mathcal{G} is defined as unique, the propagation step has already been performed while creating \mathbf{g}_1 and \mathbf{g}_2 on \mathcal{G} . The coherency is therefore guaranteed.

3.2.2 Two reference data models

This approach makes possible the application of the *Stick* operator in the case of merging data coming from two independent data models (e.g., interoperability). The only requirement is a previous alignment of their ontologies. In this case, the link between the two reference graph models is carried out using the following signature:

$$Stick(\mathcal{G}_1, \mathbf{g}_1, \mathcal{G}_2, \mathbf{g}_2) \rightarrow (\mathcal{H}, \mathbf{h})$$

with \mathcal{H} defined as the result of the ontological union (\cup_o) of $\mathcal{G}_1, \mathcal{G}_2$ respectively reference graph of $\mathbf{g}_1, \mathbf{g}_2$ and \mathbf{h} the result graph of the *Stick* operator.

In this case, the propagation step, realised one time in the previous case, must be iterated in \mathcal{H} since data models \mathcal{G}_1 and \mathcal{G}_2 were initially independent. The operational steps are the followings:

- Computation of $\mathcal{H} = \mathcal{G}_1 \cup_o \mathcal{G}_2(eq_1)$.
- Propagation using rules \mathbb{R}_p on \mathcal{H} .

3.2.3 Generalisation of these two approaches: the *Stick* operator

The reference graph \mathcal{H} corresponds (on the basis of a unique reference model or on the basis of two reference models), respectively to the graph \mathcal{G} or to $\mathcal{G}_1 \cup_o \mathcal{G}_2$ plus the propagation.

So, we can generalise the *Stick* operator between \mathbf{g}_1 et \mathbf{g}_2 by:

$$Stick(\mathcal{H}, \mathbf{g}_1, \mathbf{g}_2) \rightarrow (\mathcal{H}, \mathbf{h})$$

with \mathbf{h} , the result graph.

Operators defined on the reference data model graph are closed (i.e., the result of the application of an operator can be used as an operand for a new application of an operator). The main interest of the *Stick* operator is reinforced in the case of manipulation of attributes defined in intension. (i.e., the value of an instance is the result of a calculus function).

The definition of a calculated attribute may rely on the use of other attributes and/or ISO 13249 spatial functions (e.g., ST_Area). Operators that decrease the number of attribute (e.g., *ZoomIn*) may lead an attribute to be classified as 'FROZEN'. Attributes involved in the calculated function are no more available. The main effect is the transformation of the notion of view into the notion of snapshot (i.e., the variation of a value, due to an update of operands, is no more possible). The calculus function may potentially have all required attributes since information are added by the *Stick* operator (i.e., attributes that were required to evaluate the function).

The *Stick* operator allows us to move from the PP property of the Hermann's classification, reduced to the equality to the PP property on the basis of \subseteq . This is due to

the fact that attributes (defined in intension) such as their calculus functions were not computable may become computable by the addition of attributes (via the *Stick* operator). The classification of FROZEN attribute can therefore disappear. The ontological union of graphs maintains the original attributes by (*eq*₁).

The second property in the Hermann's classification (FPP) is guaranteed but in only one case: the absence of attributes defined in intension. The reason is that the independence is no more guaranteed in the case of a calculus function using attributes provided by the two models. In case of an attribute in the model m_2 (i.e., the addition of the only missing attribute in order to evaluate an attribute defined in intension in the m_1 model), its addition modifies the interpretation of m_1 .

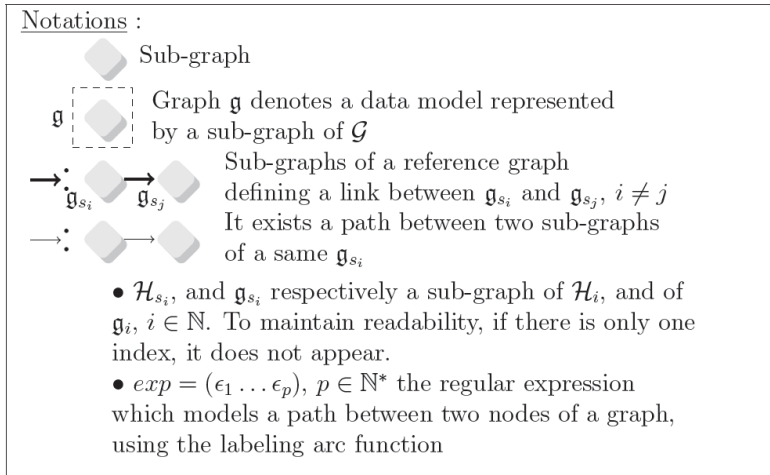
The third property in the Hermann's classification (CP) is guaranteed since it was based on coherent initial models. Attributes that do not interact in calculus functions still maintain the independence of the two data models and those that interact in the calculus function may not change anything if the calculus function is still un-computable, or make it computable but coherent with the PP property.

So, if the *ZoomIn* operator may transform an attribute to the FROZEN classification since the overall attributes to evaluate the calculus function are not anymore available, the *Stick* operator may render these attributes available and possible such a calculus.

3.3 Presentation of the formalism

The definition of the *Stick* operator requires the definition of labelling rules for nodes and arcs of \mathfrak{h} (labelling functions and characteristics of attributes in the reference associated data model).

Figure 6 Caption of figures



This article focuses on the description of the *Stick* operator applied on two graphs $\mathfrak{g}_1 = (N_1, E_1, \psi_1, v_1, \epsilon_1)$, $\mathfrak{g}_2 = (N_2, E_2, \psi_2, v_2, \epsilon_2)$ depending on two hypotheses. We

illustrate these hypotheses starting from a reference data model presented Figure 4, the associated figures use the caption defined in Figure 6.

Within the first hypothesis, these graphs are independent, that means: $g_1 \cap g_2 = \emptyset$.

Within the second hypothesis, it may exist a chain between g_1 and g_2 in \mathcal{H} (i.e., from a common ancestor to) and the lack of path between g_1 and g_2 or g_2 and g_1 (i.e., the graph is quasi-strongly connected) or there exists at least one path between a node of g_1 (resp. g_2) and a node of g_2 (resp. g_1) in \mathcal{H} .

4 Independent graphs

With independent graphs, two configurations with respect to the reference graph \mathcal{H} are possible. In the first configuration, a common ancestor node, defined between a node of g_1 and a node of g_2 exists, in the second one it does not exist.

4.1 Independent graphs: conceptual link between g_1 and g_2 [C1]

[C1] is built assuming hypotheses of Table 1. The *Stick* operator is applied between two independent graphs (i.e., g_1 and g_2) linked by a common concept in the reference graph (i.e., they conceptually have a link). In the example Figure 7, a stick operation is applied between the graph g_1 (which includes the nodes *Cinema* and *Thematic*) and the graph g_2 (which includes the nodes *Museum*, *Section* and *Art work*). The first step is to look for a common ancestor in \mathcal{H} which allows connecting these two graphs (in our case the node *City*). To simplify the presentation without loss of generality, the node *City* is connected to g_1 and g_2 by a single arc (vs. a path).

Table 1 Hypotheses of [C1]

Hypotheses
<ul style="list-style-type: none"> $g_1 \cap g_2 = \emptyset$ \nexists path between a node of g_1 (resp. g_2) and a node of g_2 (resp. g_1) \exists a quasi-strongly connected sub-graph of \mathcal{H} with g_1 and g_2

4.1.1 Node(s)

The aim is to identify in the reference graph, \mathcal{H} , a shared concept called common ancestor node between at least one node of g_1 and one node of g_2 . Let the function lg be defined as: $\mathcal{G} \times N \times N \rightarrow \mathbb{N}$. The function lg represents here the number of arc in a path between two nodes in a graph (its value is 0 if a path does not exist). In the general case, lg can be based on context dependent metrics. The function is customisable according to the application domain (e.g., social networks, spatial analysis).

This is equivalent to look for a quasi-strongly connected sub-graph in \mathcal{H} . A node with this property and which minimises the distance (number of arc according to the

definition of lg may not be unique since the reference graph may be a general graph (i.e., not a tree). The set of ancestor nodes is denoted by A (Figure 8).

$$A = \{a \in N_{\mathcal{H}}, a \notin N_2 \wedge \exists n_1 \in N_1 \mid \min_{n_i \in N_1} lg(\mathcal{H}, a, n_i) > 0 \wedge \exists n_2 \in N_2 \mid \min_{n_i \in N_2} lg(\mathcal{H}, a, n_i) > 0\}.$$

Figure 8, the common ancestor is not unique $A = \{a_1, a_2\}$. Let D_1 (resp. D_2) be the set of entering nodes in the graph \mathfrak{g}_1 (resp. \mathfrak{g}_2) in a path initiated with a node of A . D_1 (resp. D_2) is the set of nodes which are connected via a minimum length path with (at least) one common ancestor. As several paths may be defined from the same ancestor to several nodes of \mathfrak{g}_1 (resp. \mathfrak{g}_2) this set may also not be reduced to a singleton (Figure 9):

$$D_1 : \{n_i \in N_1 \mid \min_{n_i \in N_1} lg(\mathcal{H}, a, n_i) > 0, a \in A\}$$

$$D_2 : \{n_i \in N_2 \mid \min_{n_i \in N_2} lg(\mathcal{H}, a, n_i) > 0, a \in A\}$$

For example, Figure 9, $D_1 = \{n_1, n_{11}\}$ since $lg(\mathcal{H}, a_1, n_1)$ and $lg(\mathcal{H}, a_1, n_{11})$ are equal. Applied to Figure 7, D_2 is not reduced to a singleton and $A = \{City\}$, $D_1 = \{Cinema\}$ and $D_2 = \{Museum, Art work\}$.

Figure 7 Example of sub-graph in \mathcal{H} (see online version for colours)

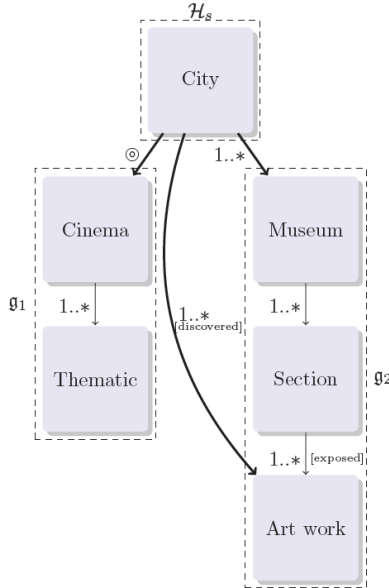


Figure 8 $A = \{a_1, a_2\}$, set of ancestors, is not reduced to a singleton (see online version for colours)

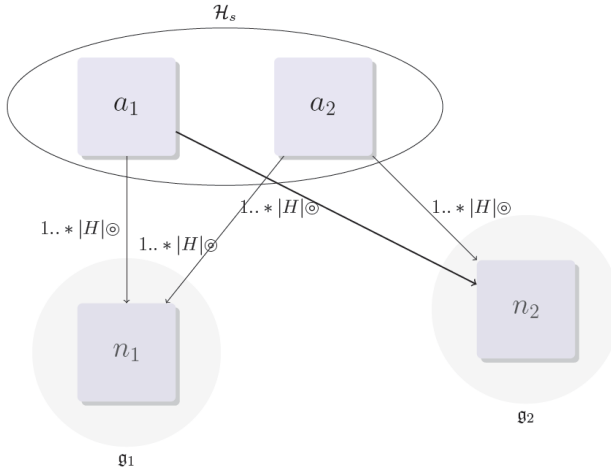
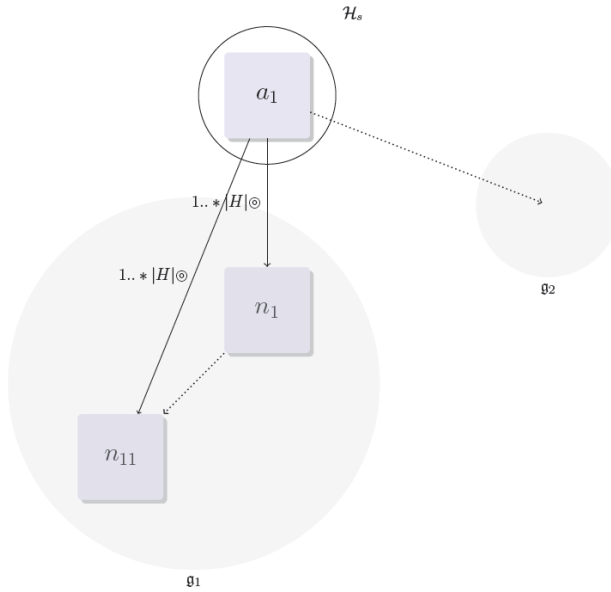


Figure 9 $D_1 = \{n_1, n_{11}\}$, set of entering nodes of g_1 , is not reduced to a singleton (see online version for colours)

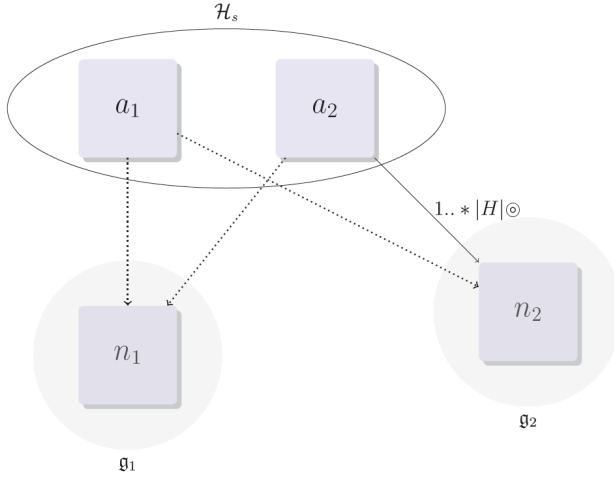


If lg is too long, concepts modeled by nodes in A may not bring relevant information according to the applied stick operation. For example, if we try to align information about cinemas in a city and those related to museum, information's about European Union which could be the abstract level of the closer common ancestor are not relevant. The function lg provides a proximity measure between graphs implied in the stick operation. It gives the degree of correlation between concepts in g_1 and g_2 .

4.1.2 Arcs: definition

The link between the graph g_1 (resp. g_2) and this common ancestor (called a) can be an arc defining a direct connection (D), or a path defining an indirect connection (I).

Figure 10 Illustration of connections II and ID (see online version for colours)



The Cartesian product of connections defines DD, DI, ID, II in \mathcal{H} . In case of a forest, direct connection is always favoured (in particular in the choice of common ancestors). Let $a \in A$, $n_1 \in D_1$ and $n_2 \in D_2$, possible connections are defined as follow:

$$\text{DD} : lg(\mathcal{H}, a, n_1) = lg(\mathcal{H}, a, n_2) = 1$$

$$\text{DI} : lg(\mathcal{H}, a, n_1) = 1 \wedge lg(\mathcal{H}, a, n_2) > 1$$

$$\text{ID} : lg(\mathcal{H}, a, n_1) > 1 \wedge lg(\mathcal{H}, a, n_2) = 1$$

$$\text{II} : lg(\mathcal{H}, a, n_1) > 1 \wedge lg(\mathcal{H}, a, n_2) > 1$$

Figure 8 illustrates the case DD. It shows labelling possibilities for arcs leaving.

$$a_1 : e_1 = (a_1, n_1) \mid n_1 \in D_1 \text{ and } e_2 = (a_1, n_2) \mid n_2 \in D_2.$$

Figure 10 shows cases II and ID (resp. DI), the dashed arc represents a path (i.e., a sequence of arcs). To simplify the presentation without loss of generality, we illustrate configurations with a unique connection (the case of a forest is a repetition of presented principles).

4.1.3 Arcs: labelling

Regardless the sub-case (DD, DI, ID II) a regular expression as defined in Figure 6 represents the labelling of a path between A and D_i . The paths (potentially reduced to an arc) are independently processed because g_1 and g_2 are independent (an action on g_1 does not have any consequence on g_2 and reciprocally, because $g_2 \cap g_1 = \emptyset$ by definition). Dealing with DI is equivalent to the application of D in DD for the direct connection, and I in II for the indirect connection.

Table 2 labelling rules (\mathbb{R}_{et})

$\{\odot\} \vee \{H, \odot\} \rightarrow \odot$
$\{H\} \rightarrow H$
$\{1..*\} \vee \{1..*, \odot\} \vee \{1..*, H\} \vee \{1..*, H, \odot\} \rightarrow 1..*$

The fundamental principle in the construction of h is a graph with graphs g_1 and g_2 and an addition of nodes in A . The connection is defined with an arc (set oriented notion of the *Stick* operator). In any case, a label for this arc (a_i, n_j) where $a_i \in A$ and $n_j \in D_k$, $k \in \{1, 2\}$ has to be defined. The labelling definition is trivial if the arc is direct, the arc is directly derived (source, destination and label) from the reference graph \mathcal{H} . In the case of a path, a regular expression (built with labels of arcs and processed as a set) models the path. The computation rules (\mathbb{R}_{et}) are presented in Table 2. These rules imply that if the regular expression contains only spatial inclusions \odot (possibly with H whatever the order is), the label associated with the arc is \odot . If the regular expression contains only H , the label is H . Finally, as soon as the symbol $1..*$ appears in the regular expression, then the label is $1..*$.

4.1.4 Result graph: definition

The result graph (i.e., h) is defined as follow:

- $N_h = N_1 \cup N_2 \cup A$
- $E_h = E_1 \cup E_2 \cup \{e_q = (a_i, n_j) \mid q \in \mathbb{N}, q > |E_1| + |E_2|, a_i \in A \wedge n_j \in D_k \mid k \in \{1, 2\} \wedge lg(\mathcal{H}, a_i, n_j) \neq 0\}$. The condition $lg(\mathcal{H}, a_i, n_j) \neq 0$ takes into account only couples involved in a path in \mathcal{H} .

The case of indirect connection (I), $lg(\mathcal{H}, a_i, n_j) > 1$, is related to the fact that n_j does not belong to direct successors of a_i , $\epsilon(e_q)$ is defined from the regular expression $exp = (\epsilon_1 \dots \epsilon_p)$, $p \in \mathbb{N}$, according to rules defined in Table 2 with:

$$\epsilon(e_q) = \odot \text{ if } \forall \epsilon_l \in exp, 1 \leq l \leq p, \epsilon_l = (\odot \vee H) \wedge \exists l \mid \epsilon_l = \odot$$

$$\epsilon(e_q) = H \text{ if } \forall \epsilon_l \in \text{exp}, 1 \leq l \leq p, \epsilon_l = H$$

$$\epsilon(e_q) = 1..* \text{ if } \exists \epsilon_l \in \text{exp}, 1 \leq l \leq p \mid \epsilon_l = 1..*$$

4.1.5 Result graph: dealing with potential redundancy

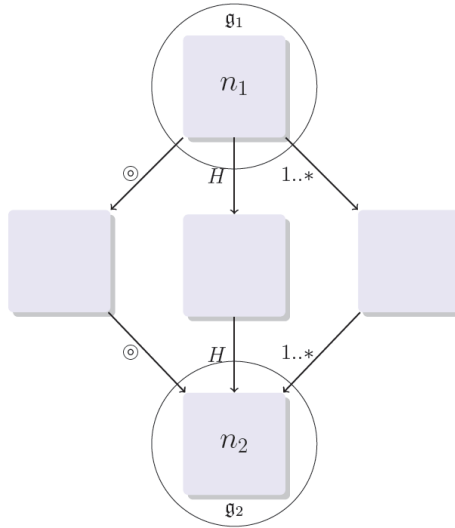
Two types of redundancy may appear. One is due to the way the result graph is built, and the second one is related to the application of propagation rules.

Table 3 Choice of a relevant label

<i>arc (source, destination, label)</i>	(n_o, n_d, \odot)	$(n_o, n_d, 1..*)$	(n_o, n_d, H)
(n_o, n_d, \odot)	-	\odot	\odot
$(n_o, n_d, 1..*)$		-	H
(n_o, n_d, H)			-

While building the set of arcs $E_{\bar{h}}$, as soon as it exists at least two paths in \mathcal{H} such as $lg(\mathcal{H}, a_i, n_j)$ are equal, as much arcs are generated (Figure 11). If these arcs have the same label, duplicate entries are deleted due to the set property. If the labels are different then rules of Table 3 have to be applied in order to choose the relevant label for the added arc in $E_{\bar{h}}$. The idea is to keep the maximum information for end users. $1..*$ (resp. H) is less accurate than \odot which implies a spatial inclusion whereas $1..*$ only implies a simple relationship (resp. sub-class information H , we favour spatial relationships).

Figure 11 Case of multiples arcs (see online version for colours)



The application of propagation rules on \mathcal{H} can imply that the application of the *Stick* operator creates redundancies at the data model level, because g_1 et g_2 are independents. In this case, in order to delete these redundancies, we apply the Algorithm 1 on each node of A . The principles of the algorithm are: for each node in D_1 and D_2 related to a common ancestor in A with an arc labelled with H , if we have an identical attribute (in each node), this attribute is moved to the common ancestor and deleted from the studied nodes (of D_1 and D_2). It is not possible to do an equivalent process for another label than H because the attribute characteristic is likely to be degraded (GLOBAL vs. SUBSET).

Algorithm 1 Dealing with redundancies: *Cleaning* (h, a, D_1, D_2)

```

//  $D'_1$ (resp.  $D'_2$ ): subset of nodes of  $D_1$  (resp.  $D_2$ ) such as the arc label between an
ancestor  $a \in A$  and a node of this set is  $H$  in the graph

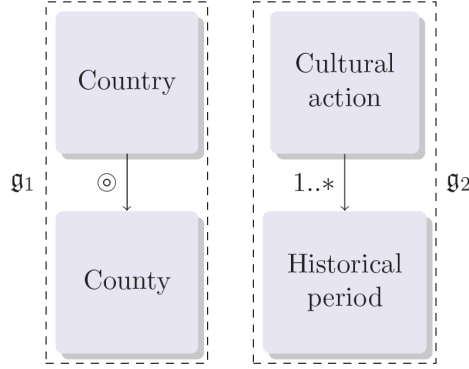
Let  $D'_1 = \{n \in D_1 \mid e_h(a, n) = H\}$ 
Let  $D'_2 = \{n \in D_2 \mid e_h(a, n) = H\}$ 

for all attribute  $a_u$  of  $v(n) \mid n \in D'_1$  do
    //  $D'_2$ : to take it into account is useless because  $a_u$  has also to belong to  $D'_1$ 
    //if the attribute exists in each node of  $D'_1$  (resp.  $D'_2$ )
    if  $(\forall n_1 \in D'_1, a_u \in v(n_1) \wedge \exists e_1 = (a, n_1) \in E_h) \wedge (\forall n_2 \in D'_2, a_u \in v(n_2) \wedge \exists e_2 = (a, n_2) \in E_h)$  then
        //it is possible to delete  $a_u$  in child nodes, and to rise it to the ancestor node  $a$ 
         $v(n_1) = v(n_1) - a_u$ 
         $v(n_2) = v(n_2) - a_u$ 
         $v(a) = v(a) \cup a_u$ 
    end if
end for

```

4.2 Independent graphs: no conceptual link between g_1 and g_2 [C2]

[C2] is built assuming hypotheses of Table 4. The *Stick* operator is applied between two independent graphs. In the example presented Figure 12, a stick operation is applied between the graph g_1 (which includes nodes *Country* and *Country*) and the graph g_2 (which includes nodes *Cultural action* et *Historical period*). Under assumptions of Table 4, it does not exist in the reference graph \mathcal{H} a shared concept between at least one node of g_1 and one node of g_2 . So, the process cannot be totally automated because we have no inner information that could link a graph g_1 to a graph g_2 . Conceptually the *Stick* operator does not force the connectivity of the result graph. As we require the closure of operators, and as the *ZoomIn* operator needs a connected graph, so we impose a connected graph as a result. The *Stick* operator aims to show a link between those two graphs. To deal with this case, firstly we propose an automated phase which creates a consistent result graph. Then an interactive phase, with an end-user, will precise the semantic data model.

Figure 12 Example of sub-graph in \mathcal{H} (see online version for colours)**Table 4** Hypotheses of [C2]

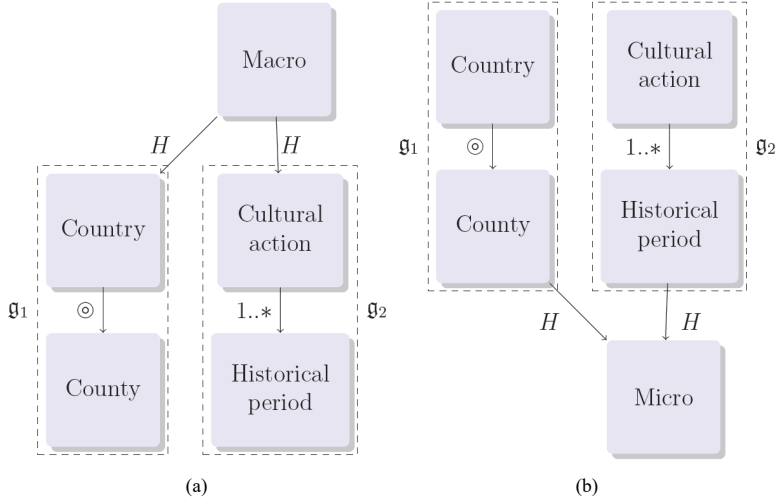
Hypotheses
<ul style="list-style-type: none"> • $g_1 \cap g_2 = \emptyset$ • \nexists path between a node of g_1 (resp. g_2) and a node of g_2 (resp. g_1) • \nexists a quasi-strongly connected sub-graph with g_1 and g_2 in \mathcal{H}

4.2.1 Node(s)

We propose the addition of a node which plays the role of ‘shared concept’, let n_o be this node, and $A = \{n_o\}$. So, $N_b = N_1 \cup N_2 \cup A$. According to a end user’s needs (interactive phase), this node can be integrated using two approaches. Either it allows the definition at a macro level [Figure 13(a)], or at a micro level [Figure 13(b)]. At a macro level, we use roots of the graph g_1 (resp. g_2). These roots exist because the graph is without circuit, sets of root nodes are defined by $D_{r1} = \{n \in N_1 \mid \Gamma^-(n) = \emptyset\}$ in g_1 and $D_{r2} = \{n \in N_2 \mid \Gamma^-(n) = \emptyset\}$ in g_2 . In the example [Figure 13(a)] $D_{r1} = \{Country\}$ and $D_{r2} = \{Cultural\ action\}$. A macro level node added to the data model, [Figure 13(a)], would be an entity which represents an international event in order to justify the link between *Country* et *Cultural action* (e.g., universal exhibition).

At the micro level, we deal with leaves of the graph g_1 (resp. g_2). These leaves exist for the same reasons as at the macro level, and there are defined in two sets $D_{l1} = \{n \in N_1 \mid \Gamma^+(n) = \emptyset\}$ in g_1 and $D_{l2} = \{n \in N_2 \mid \Gamma^+(n) = \emptyset\}$ in g_2 . In the example [Figure 13(a)] $D_{l1} = \{County\}$ and $D_{l2} = \{Historical\ period\}$. A micro level node added to the data model, [Figure 13(b)], would be an entity which represents local events like the celebrations of WW2 Normandy landing birthday (*Historical period*: second world war, *County*: Normandy).

Figure 13 [C2]: two levels for a stick operation, (a) macro level (b) micro level (see online version for colours)



Whatever the level (macro exclusive or micro) is, a end user will precise attributes that (s)he would like to define, the visibility of these attributes (i.e., public or private) and their characteristics (i.e., GLOBAL, SUBSET or FUZZY)¹.

4.2.2 Arcs: definition

In order to ensure the connectivity of the result graph, we need to establish links between the graph g_1 and the graph g_2 . We define arcs with H label from n_o to each node in sets D_{r1} and D_{r2} for the macro level (resp. from D_{l1} and D_{l2} to n_o for the micro level).

4.2.3 Result graph: definition

The graph result (i.e., h) is defined as follow:

- $N_h = N_1 \cup N_2 \cup A$
- $E_h = E_1 \cup E_2 \cup \{e_q = (n_o, n_j) \mid q \in \mathbb{N} \wedge q > |E_1| + |E_2|, n_j \in D_{rk} \mid k \in \{1, 2\}\}$ for the macro level (resp. $e_q = (n_j, n_o) \mid q \in \mathbb{N} \wedge q > |E_1| + |E_2|, n_j \in D_{lk} \mid k \in \{1, 2\}$ for the micro level)
- $\epsilon(e_q) = H$

4.2.4 Result graph: dealing with potential redundancy

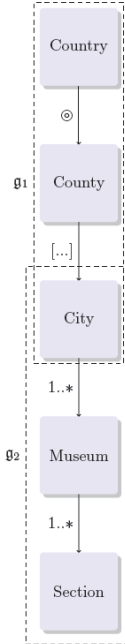
Considering the addition of this information which did not exist before the application of the *Stick* operator, there cannot exist any redundancy at the micro level. For the macro

level, redundancies may appear because g_1 and g_2 are independent. We are facing a similar configuration to [C1]. A is defined by the added node $A = \{n_o\}$. D_{r1} (resp. D_{r2}) is the set that includes the roots of g_1 (resp. of g_2). In this configuration, we can apply the Algorithm 1, *Cleaning* (h, n_o, D_{r1}, D_{r2}).

5 Non-independent graphs

The loss of independency can be materialised by two configurations. The first configuration is defined by shared nodes/arcs between g_1 and g_2 . The second one is defined by the existence of a path in the reference graph \mathcal{H} , from g_1 (resp. g_2) to g_2 (resp. g_1).

Figure 14 [C3]: a node in common (see online version for colours)



5.1 Non-independent graphs: shared node(s) and/or arcs [C3]

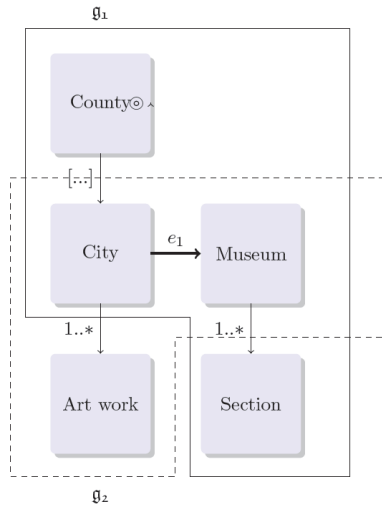
[C3] is built assuming hypotheses of Table 5. This case shows the *Stick* operator between two graphs g_1 and g_2 where there is at least one node (and/or arc(s)) which belongs to the two graphs. In the example Figure 14, a stick operation is applied between the graph

g_1 (which includes nodes: *Country*, *County* and *City*) and the graph g_2 (which includes nodes: *City*, *Museum* and *Section*). In this example, only the node *City* is in common. Other contexts can appear like the one described Figure 16 where an arc (i.e., e_1) implies two nodes (*City*, *Museum*) in common between g_1 and g_2 .

Table 5 Hypotheses of [C3]

Hypotheses
<ul style="list-style-type: none"> $g_1 \cap g_2 \neq \emptyset$

Figure 15 [C3]: an arc and two nodes in common (see online version for colours)



5.1.1 Node(s)

According to hypotheses defined in Table 5, it exists at least one node that belongs to the two sets N_1 and N_2 . The characteristics defined for the attributes of common nodes can be different (i.e., they may come from the application of operators which leads to different propagation results). Our objective is to keep the maximum of information about the spatialisation of concerned attributes (as we do when dealing with multiple arcs of [C1] Table 3). Therefore, we apply rules presented in Table 6 which favour the characteristic SUBSET against the others (GLOBAL and FUZZY). The characteristic GLOBAL is favoured against FUZZY.

Table 6 Choice of characteristics to be favoured in case of attribute redundancy

	SUBSET	GLOBAL	FUZZY
SUBSET	-	SUBSET	SUBSET
GLOBAL	SUBSET	-	GLOBAL
FUZZY	SUBSET	GLOBAL	-

5.1.2 Arcs: definition

According to hypotheses defined in Table 5, arcs which come from or arrive to common node(s), are already present, either in E_1 , or in E_2 , or in E_1 and E_2 . Depending on the semantic linked to graphs g_1 and g_2 , an arc, e_i , can be present in the two graphs but with a different labelling. Therefore, we are facing to the same redundancy as in case [C1]. We process it in a similar way with Table 3. For example, application of rules in Table 2 or the composition of operations, like the *ZoomIn*, may lead to this kind of configuration.

5.1.3 Result graph: definition

The result graph (i.e., h) is defined as follow:

- $N_h = N_1 \cup N_2$
- $E_h = E_1 \cup E_2$

5.1.4 Result graph: dealing with potential redundancy

According to the result graph construction, no redundancy can appear in the result data model because the union operator is a set-oriented operator, therefore it deletes all duplications [node(s) and arc(s)].

5.2 Non-independent graphs: existence of path(s) between g_1 and g_2 [C4]

If there are paths between the graph g_1 (resp. g_2) and g_2 (resp. g_1), possibly both, an ambiguity in the interpretation may appear. Two paths can be defined from the same couple (source node, destination node) but they do not model a semantic which leads to the same instances. Paths between *City* and *Art work* in Figure 4 are an example of such a configuration. An art work may be exhibited in a city different from the one where it was discovered. We distinguish two sub-cases:

- a without considering instances (model level)
- b by taking into account instances (instance level).

5.2.1 [C4]a: without considering instances

[C4]a is built according to hypotheses defined in Table 7. This case shows the *Stick* operator between two graphs g_1 and g_2 such as it exists a path in \mathcal{H} between them, but they do not share any node. In the example Figure 15, a *Stick* operation is applied

between the graph g_1 (which includes nodes: *Country* and *County*) and the graph g_2 (which includes nodes: *Museum* and *Section*). In this example, there is a path in \mathcal{H} between the node *County* in g_1 and the node *Museum* in g_2 .

Figure 15 [c4]a: example (see online version for colours)

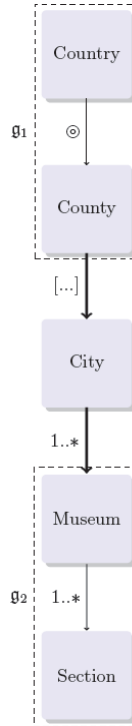


Table 7 [C4]a: hypotheses

-
- $g_1 \cap g_2 = \emptyset$
 - \exists a path between a node of g_1 and a node of g_2 (possibly reciprocally)
-

Table 8 [C4]b: hypotheses

-
- $g_1 \cap g_2 = \emptyset$
 - \exists a path between two node of g_1 and one of these paths uses node(s) in g_2 (and possibly reciprocally)
-

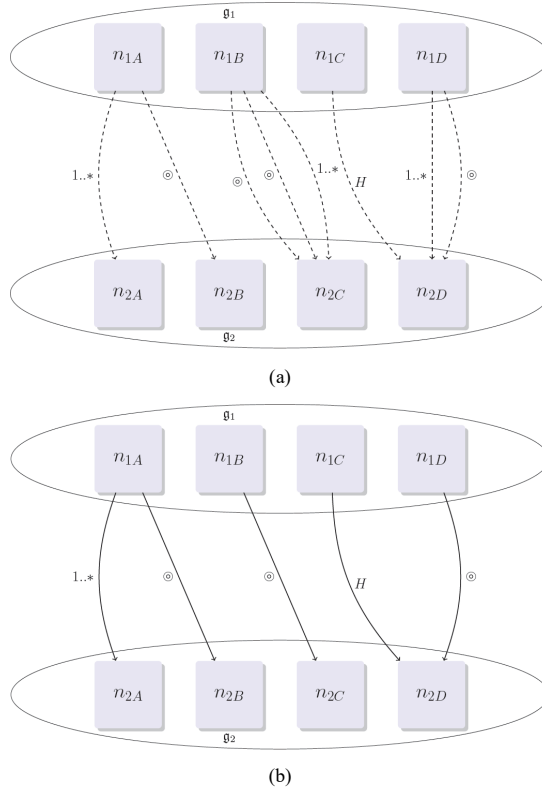
Table 9 Synthesis of application cases for *Stick* operator

Graph independence	Hypotheses	Result graph	Management of redundancy
<ul style="list-style-type: none"> $g_1 \cap g_2 = \emptyset \nexists$ path between a node of g_1 (resp. g_2) and a node of g_2 (resp. g_1) in \mathcal{H} 	<p>[C1]</p> <ul style="list-style-type: none"> Quasi-strongly connected sub-graph of \mathcal{H} with g_1 and g_2 	$N_0 = N_1 \cup N_2 \cup A$ $N_0 = E_1 \cup E_2 \cup \{e_q = (a, n_i) \mid q \in \mathbb{N}, q > E_1 + E_2 , a \in A \wedge n_j \in D_k / k \in \{1, 2\} \text{ and } lg(\mathcal{H}, a, n_i) \neq 0\}$ Label case of indirect link: Table 3	Multiples arcs: Table 4 attributes: Algorithm 1 Cleaning (b, a, D_1, D_2) Only macro level attributes: Algorithm 1 Cleaning (b, n_o, D_{i1}, D_{r2})
	<p>[C2]</p> <ul style="list-style-type: none"> \nexists quasi-strongly connected sub-graph of \mathcal{H} with g_1 and g_2 	$N_0 = N_1 \cup N_2 \cup A$ $E_0 = E_1 \cup E_2 \cup \{e_q = (n_o, n_i) \mid q \in \mathbb{N} \wedge q > E_1 + E_2 , n_j \in D_k / k \in \{1, 2\}\}$ for the macro level $eq = (n_j, n_o) \mid q \in \mathbb{N} \wedge q > E_1 + E_2 , n_j \in D_k / k \in \{1, 2\}$ for the micro level and $\epsilon(e_q) = H$	Cleaning $(b, n_o, a, D_{r1}, D_{r2})$ No redundancy possible by construction Attributes: Algorithm 2 Cleaning $(b, a, \sigma_{n=0}(\text{newLinks}))$
<ul style="list-style-type: none"> $g_1 \cap g_2 = \emptyset$ $g_1 \cap g_2 = \emptyset$ 	<p>[C3]</p> <p>cas4a (model)</p> <ul style="list-style-type: none"> \exists a path between a node of g_1 and a node of g_2 (pos. reciprocally) <p>cas4b (instance)</p> <ul style="list-style-type: none"> \exists two paths between two nodes of g_1 and one of this path uses g_2 nodes (pos. reciprocally) 	$N_0 = N_1 \cup N_2; E_0 = E_1 \cup E_2$ $N_0 = N_1 \cup N_2$ $E_0 = E_1 \cup E_2 \cup D_{g1-2} \cup D_{g2-1}$ $N_0 = N_1 \cup N_2 \cup \{n_{diff}\}$ $E_0 = E_1 \cup E_2 \cup D_{g1-2} \cup D_{g2-1} \cup \{(n_2, n_{diff}) \mid \epsilon(n_2, n_{diff}) = H\}$	

5.2.2 Nodes and arcs

According to hypotheses defined in Table 7, the principle is to generate an abstraction of the minimum path(s) (minimum in the sense of lg) between graphs g_1 and g_2 using one (or more) arc(s).

Figure 16 Dealing with redundancy for D_{g_1-2} (see online version for colours)



Unlike [C1], we have not a set of ancestors A from the graph \mathcal{H} (which contained origins of arcs).

We define a set D_{g_1-2} (resp. D_{g_2-1}) of arcs:

$$D_{g_1-2} = \{(n_o, n_d) \mid n_o \in N_1 \wedge n_d \in N_2\}$$

$$D_{g_2-2} = \{(n_o, n_d) \mid n_o \in N_2 \wedge n_d \in N_1\}.$$

Let $[]$ be the list constructor and $\{\}$ be the set constructor.

Let *Link* be the domain which represents a minimal path, *Link*: $[E]$. A minimal path, *l*, is a list of arcs $l = [e_1 \dots e_n]$.

Let *initialNode*: *Link* $\rightarrow N$ (resp. *finalNode*: *Link* $\rightarrow N$) be a function which gives initial node (resp. final node) of a minimal path.

Let *C*: $\{Link\}$ the domain which represents the set of minimal links between two graphs (i.e., $c_i \in C$, $lg(\mathcal{H}, initialNode(c_i), finalNode(c_i)) > 0$ and minimal) and the function *paths*: $\mathcal{H} \times G \times G \rightarrow C$ which generates the set of minimal paths between two graphs:

$$\text{Let } c = paths(\mathcal{H}, g_1, g_2) \text{ (res. } paths(\mathcal{H}, g_1, g_1)).$$

Let *newLinks*: $\{(n_o: N; n_d: N; label) = \{(initialNode(c_i), finalNode(c_i), \epsilon)\} | c_i \in C, 1 \leq |C|$ be the set which contains paths. Rules in Table 2 have been applied on all these paths in order to obtain a set of labelled arcs. Figure 16(a) shows the set of minimal labelled paths between g_1 and g_2 . The set *newLinks* is illustrated Figure 16(b). Redundant arcs and multiple arcs have been deleted, respectively since *newLinks* is a set and with rules of Table 3.

Let Π be the operator of projection and σ be the operator of selection.

The set D_{g1-2} (resp. D_{g2-1}) is defined by:

$$D_{g1-2} \text{ (resp. } D_{g2-1}) = \{\Pi_{N \times N}(newLinks)\}.$$

In the example Figure 15, $D_{g1-2} = \{(County, Museum)\}$ and $D_{g2-1} = \emptyset$.

5.2.3 Result graph: definition

The result graph (i.e., h) is defined as follow:

- $N_h = N_1 \cup N_2$
- $E_h = E_1 \cup E_2 \cup D_{g1-2} \cup D_{g2-1}$

5.2.4 Result graph: dealing with potential redundancy

By construction of sets D_{g1-2} and D_{g2-1} , they do not contain redundancy and multiple arcs.

Redundancies that would appear at the attribute level are deleted by application of Algorithm 2. The philosophy of this algorithm is similar to the one of Algorithm 1. Nodes which have to be computed are defined by $D' = \Pi_{no}(newLinks)$. For each node *a* of D' , the Algorithm 2 *Cleaning*(*h*, *a*, $\sigma_{no} = {}_a(newLinks)$) is applied.

Algorithm 2 Dealing with redundancy: *Cleaning*(*h*, ancestor, arcsFromAncestor)

```

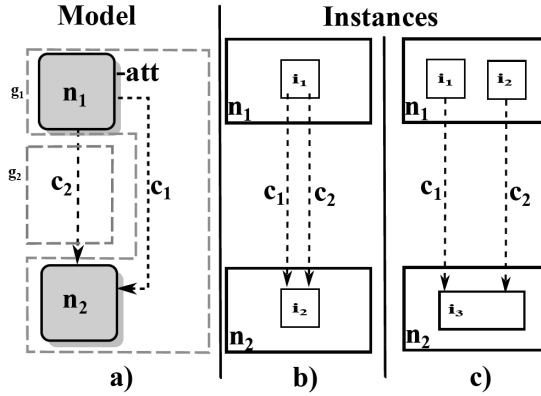
if ( $\forall(\text{ancestor}, n_d, \epsilon) \in \text{arcsFromAncestor} \Rightarrow \epsilon = H$ ) then
  if ( $\exists att | \forall(\text{ancestor}, n_d, \epsilon) \in \text{arcsFromAncestor } a_{att} \in v(n_d)$ ) then
    //we can delete  $a_{att}$  in child nodes, and ascent it in the ancestor
     $v(n_d) = v(n_d) - a_{att}$ 
     $v(\text{ancestor}) = v(\text{ancestor}) \cup a_{att}$ 
  end if
```

end if

5.2.5 [C4]b: considering instances

The general issue is illustrated Figure 17. [Figure 17(a)], a node n_1 of g_1 is linked to a node n_2 of g_1 by two paths (e.g., c_1 and c_2). As a result of a stick operation, one of these paths, c_2 , involves nodes/arcs in g_2 . If the semantic leads to link the same instance by c_1 and c_2 , we are facing the case [C4]a [Figure 17(b)]. However [Figure 17(c)], if the semantics leads to link i_1 and i_2 acting as two instances of n_1 , it is necessary to deal with alphanumeric information associated with these two instances (i.e., i_1, i_2).

Figure 17 General issue



In the example in Figure 18, a stick operation is applied between the graph g_1 (which includes nodes: *City*, *Art work*) and g_2 (which includes nodes: *Museum*, *Section*). For example a city (i_1) has a museum and art works are exposed in this museum. An art work (i_3) has been discovered in a different city (i_2 by c_1), than the one where it is exposed (i_1 by c_2). This stick operation leads to two paths between *City* and *Art work*. In [C4]b, we study the propagation of attributes at an instance level, because two different instances of *City* (i.e., n_1) are linked to a same instance of *Art work* (i.e., n_2).

5.2.6 Nodes

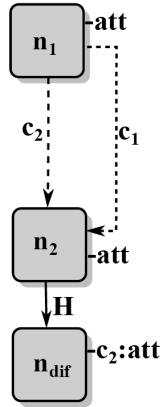
Conceptually, there are two types of node (Figure 19):

- 1 the nodes where no duplication of instance appears, the initial node of g_1 (resp. g_2) keeps its role
- 2 the nodes where this duplication exists, it is necessary to distinguish information from the first instance (the initial node) and from the second one. Names of the

5.2.9 Result graph: dealing with potential redundancy

The management of redundancies is similar to [C4]a.

Figure 19 Application



6 Conclusions

The tremendous increasing of services on different media (e.g., in vehicles, on mobile phones), for example with navigation aid tools is now a reality. The provision of semantic information to supplement cartographic representations becomes an important issue. Nowadays, commercial approaches are focused on cartographic representation of navigations for example, and the provision of factual information's (i.e., conventional alphanumeric data) is still an open problem. We propose in this article a contribution in this direction. We propose a *Stick* operator by defining a set of rules in order to guarantee a constant spatial coherency within a data model.

The *Stick* operator gives additional information by integrating into a same user's data model, information from different points of view. These points of view may belong to the same information system or may be provided by multiple sources (e.g., interoperability). We require that in the case of multiple sources, an ontology alignment has already been performed, in order to avoid any interpretation problems in the manipulation of data models. In this article, we start with a reference model, defined by a graph, and we define rules to be applied in order to merge two data models. Figure 10 presents a synthesis of our propositions. We can see for each case (i.e., properties checked by graphs): application hypotheses, result graph and the management of potential redundancies due to the application of the *Stick* operator. This operator is considered under two main hypotheses, dependent graphs or not, which are derived into sub-cases (i.e., linked to the existence of path(s) or not between/in the graphs to merge).

Using the formal properties defined by Herrmann et al. (2007) (CP, FPP, CP), we propose an operator that respects the following properties: binary, symmetric, closed, set-oriented, PP property, partially the FPP property (as soon as no calculated attributes are involved in the sub-data models) and the CP property.

It still remains to integrate the operator of *ZoomIn* (proposed in a previous article (Del Mondo and Mainguenaud, 2016) and the *Stick* operator into a single application, and to finish this work by the definition of a generalisation (on the semantic plan) operator, the *ZoomOut* operator, counterpart of the *ZoomIn* operator.

References

- Adnani, M.E., Yétongnon, K., Benslimane, D. and Abdelwahed, E.H. (2001) 'A multiple layered functional data model to support multiple representations and interoperability of GIS: application to urban management systems', in Aref, W.G. (Ed.): *ACM-GIS 2001, Proceedings of the Ninth ACM International Symposium on Advances in Geographic Information Systems*, ACM, Atlanta, GA, USA, pp.70–75, 9–10 November 2001.
- Aiken, A., Chen, J., Stonebraker, M. and Woodruff, A. (1996) 'Tioga-2: A direct manipulation database visualization environment', in *Proceedings of the Twelfth International Conference on Data Engineering, ICDE '96*, IEEE Computer Society, Washington, DC, USA, pp.208–217.
- Del Mondo, G. and Mainguenaud, M. (2016) Dynamic multi-granularity semantic representations for navigation facility services, *Int. J. of Spatial, Temporal and Multimedia Information Systems*, Vol. 1 No. 1, pp.3–29.
- Cruz, I. F. and Xiao, H. (2008) 'Data integration for querying geospatial sources', in Sample, J., Shaw, K., Tu, S. and Abdelguerfi, M. (Eds.): *Geospatial Services and Applications for the Internet*, pp.110–134, Springer.
- Cruz, I.F., Sunna, W. and Chaudhry, A. (2004) *SEMI-Automatic Ontology Alignment for Geospatial Data Integration*, of *Lecture Notes in Computer Science*, Springer, Vol. 3234, pp.51–66.
- Estublier, J., Ionita, A.D. and Nguyen, T. (2008) 'Code generation for a bidimensional composition mechanism', in Huzar, Z., Koci, R., Meyer, B., Walter, B. and Zendulka, J. (Eds): *Software Engineering Techniques – Third IFIP TC 2 Central and East European Conference, CEE-SET 2008*, Brno, Czech Republic, Revised Selected Papers, of *Lecture Notes in Computer Science*, Springer, Vol. 4980, pp.171–185, 13–15 October 2008.
- Fonseca, F.T., Egenhofer, M.J., Davis, C.A. and Câmara, G. (2002) 'Semantic granularity in ontology-driven geographic information systems', *Ann. Math. Artif. Intell.*, Vol. 36, Nos. 1/2, pp.121–151.
- Harel, D. and Rumpe, B. (2004) 'Meaningful modeling: What's the semantics of 'semantics'?', *Computer*, Vol. 37, No. 10, pp.64–72.
- Herrmann, C., Krahn, H., Rumpe, B., Schindler, M. and Völkel, S. (2007) 'An algebraic view on the semantics of model composition', in *Proceedings of the Third European Conference on Model Driven Architecture – Foundations and Applications*, pp.99–113.
- Hoel, E.G., Bakalov, P., Kim, S. and Brown, T. (2015) 'Moving beyond transportation: utility network management', in Bao, J., Sengstock, C., Ali, M., Huang, Y., Gertz, M., Renz, M. and Sankaranarayanan, J. (Eds.): *SIGSPATIAL '15*, ACM.
- Lieberman, J., Singh, R. and Goad, C. (2007) *W3C Geospatial Ontologies*, Report, W3C [online] <http://www.w3.org/2005/Incubator/geo/XGR-geo-ont/> (accessed 7 July 2017).
- Longo, J.S.C. and Medeiros, C.B. (2013) 'Providing multi-scale consistency for multi-scale geospatial data', in *Conference on Scientific and Statistical Database Management, SSDBM '13*, Baltimore, MD, USA, pp.1–12, 29–31 July 2013.
- Mustière, S. and Devogele, T. (2008) 'Matching networks with different levels of detail', *GeoInformatica*, Vol. 12, No. 4, pp.435–453.

- OGC-WM (2006) [online] <http://www.opengeospatial.org/standards/wms> (accessed 7 July 2017).
- Olteanu-Raimond, A. and Mustière, S. (2008) 'Data matching – a matter of belief', in *Headway in Spatial Data Handling, 13th International Symposium on Spatial Data Handling, Montpellier, France*, pp.501–519, 23–25 July 2008.
- Partyka, J., Alipanah, N., Khan, L., Thuraisingham, B. and Shekhar, S. (2008) 'Content-based ontology matching for GIS datasets', in *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS'08*, New York, NY, USA, ACM, pp.1–4.
- Pavel, S. and Euzenat, J. (2013) 'Ontology matching: state of the art and future challenges', *IEEE Trans. on Knowl. and Data Eng.*, Vol. 25, No. 1, pp.58–176.
- Stell, J. and Worboys, M. (1998) 'Stratified map spaces: a formal basis for multiresolution spatial databases', in *SDH'98 Proceedings 8th International Symposium on Spatial Data Handling, International Geographical*, pp.180–189.
- Ullman, J.D. (1988) *Principles of Database and Knowledge-Base Systems*, Vol. I, Computer Science Press, Rockville, Maryland.
- Zhao, T., Zhang, C., Wei, M. and Peng, Z-R. (2008) *Ontology-Based Geospatial Data Query and Integration*, pp.370–392, Springer Berlin Heidelberg, Berlin, Heidelberg.

Notes

- 1 Only attributes with a public visibility can be spread, the default attributed characteristic is GLOBAL (the more restrictive).