CHAPTER 6

Database Server Models for WMS and WFS Geographic Web Services

Souissi Nissrine[a] and *Mainguenaud Michel*[b]

Who is the corresponding author?

Introduction

Geographical Information Systems (GIS) have improved their services in the past years, mainly due to two factors: the overwhelming spread of Internet (e.g., speed, security, data availability) and the use of web technologies. Due to these evolutions, new GIS services have appeared. They provide the diffusion (e.g., interactive maps), the analysis or updates of geo-localized data.

These systems produce and manipulate huge amounts of geographical data (e.g., Google Earth) which is stored all over the world in various places using various formats. These formats may be closed (i.e., proprietary dependent) or opened, normalized or not. Consequently, analysis and/or decision-making may be difficult or time-consuming due to this heterogeneity of formats. To avoid these drawbacks and in order to facilitate accesses and usage of these data, the Open Geospatial Consortium (OGC) defined some propositions designed to ease the interoperability of cartographical servers thanks to interfaces defined in technical specifications

[a] Ecole Nationale de l'Industrie Minérale (ENIM)–Dept. Informatique, Avenue Hadj Ahmed Cherkaoui, B.P. 753, Agdal–10000 Rabat–MAROC.
Email: nissrine.souissi@enim.ac.ma

[b] Institut National des Sciences Appliquées de Rouen–LITIS EA4108, 685 Avenue de l'Université, B.P. 08, F76801 Saint-Etienne du Rouvray–FRANCE.
Email: michel.mainguenaud@insa-rouen.fr

(i.e., the implementations can be provided by several companies). One of these propositions is named "Web services" (OGC 2005). These propositions allow accesses to geographical data without taking into account physical aspects of communications (i.e., with an URL). They also propose a set of parameters, methods and communication rules to simplify accesses and manipulations as soon as clients and servers respect them.

The Web Map Service (WMS) (OGC 2006) and the Web Feature Service (WFS) (OGC 2010b), used in this chapter, are OGC standards for web services. WMS deals with the dynamic production of maps as images, built with geo-referenced data. WFS are designed to provide an access and a manipulation tool of geographical data within a map. End-users' interactions are reduced to a selection of an object on a map. This selection raises the GetFeatureInfo operation of WMS or the GetFeature operation of WFS and as a result these operations provide data associated with a geographical map or object. Information may be alphanumerical such as for example a town name or graphical such as for example the segments defining a border of a town.

Designing a database schema requires analyzing the information system and enumerating data which should be handled by the Data Base Management System (DBMS). The implementation of a database schema is divided into three main levels according to the ANSI/X3/SPARC recommendations (ANSI-SPARC) and (Brodie and Schmidt 1982). The most important level is the intermediate one. This level models all available information to be managed by a DBMS. The lower level defines the physical storage system depending on a set of parameters (e.g., operating system, update rate, and number of users). The upper level defines the specific views of a database schema, each of which is adapted to a specific application. Some data available in the DBMS are presented, some are hidden and others are re-organized depending on users' needs. As an example a tourist application may not be concerned by information about the number of television sets in a specific area but a marketing application is concerned by this information. In the first application, this information will be hidden and application users are not aware that this data is available in the information system. In the second case this data will be presented (may be as aggregated figures). Depending on the physical storage some data may be easy to obtain, some may require the resolution of a complex query and this process may take time.

In our case, we are in the context of distributed applications all over the world. Database administrators could not know in advance the different kinds of applications (i.e., their needs) that will query their databases. Therefore, due to the number of potential applications and users, providing the definition of external schemas as defined in the ANSI/SPARC (ANSI-SPARC) and (Brodie and Schmidt 1982) decomposition is not realistic at all.

As a consequence, end-users must be aware of the distant database schema and its semantics (with conventional problems such as aggregative / metrics units, clear definition of the scope, and semantics of the attributes). Some data available in the database schema are relevant for their needs, some are not.

GetFeatureInfo and GetFeature operations of the WMS / WFS services will query the database schema. They provide the set of data associated with a map or a geographical object. They present some limitations since they cannot query an external schema adapted to a specific user's need and no information about the time required to obtain data is available. They require the analysis of distant database schemas. Furthermore, people involved in this task are not the distant database designers since they are distant application designers. The results of WMS / WFS operations may therefore not be adapted to decision making from a specific application point of view (e.g., tourist, social, economical, transports).

In order to adapt available attributes to end-users' needs, we propose a solution that is independent of GetFeatureInfo and GetFeature operations of WMS and WFS. We still respect the normalized interface of these operations within these services. We propose an extension of the database schema in distant databases in order to precise answers that these operations may provide. The answers may be alphanumerical attributes or results of methods (i.e., functions) viewed as alphanumerical or graphical attributes. The proposed solution is to enhance the data model of distant databases with typed links on a conceptual lattice and information about the operational complexity required to obtain data. These types are defined by distant database administrators (i.e., people who have the best knowledge of available information and the required time to get these data). This solution provides a dynamic data model as a result of querying and is a semi-automatic process to define relevant data for these operations.

In the second section, we present the state of the art in the spatial enhancement of data models. In the third section, we present data model extensions suggested by us. In the fourth section, we present pre-defined rules to choose relevant attributes for these operations. In the fifth section, we present a conclusion and some perspectives.

State of the Art

We present in this section, research works devoted to enhance the semantic in geographical Web services. Two main approaches are possible: the extension of the norm or to define a service that is closer to end-users' needs.

In the first part, we present solutions promoted by the Open Geospatial Consortium (OGC 2005). In the second part, we present a model-oriented

proposition for a Web service. In the third part, we present a user-oriented database schema transformation. We end this section with a synthesis of such propositions.

OGC propositions

Among the propositions defined by the OGC, we focus on specific operations associated with Web Services: the Web Map Service (WMS) and the Web Feature Service (WFS). We particularly focus on the GetFeatureInfo and GetFeature operations available in these services.

WMS

Information provided in this part is based on the standard documentation for WMS version 1.3.0 (OGC 2006). The WMS standard induces three main operations: GetCapabilities, GetMap, and GetFeatureInfo. Figure 1 shows these operations. First of all, we present relevant operations for our topic: GetMap and GetFeatureInfo. Then we present the limits of the GetFeatureInfo operation.

The most important facility for a cartographical server is to provide a map from its available layers. Parameters such as desired layers, representation styles, size, relevant geographical areas, the projection system or the output format must be provided to the GetMap operation. The obtained result is a map (in general an image that can be displayed by conventional web browsers). This operation corresponds to a visualization process.

The GetFeatureInfo operation corresponds to a query process. This is a mandatory facility for a Webmapping application. Webmapping is a word used to represent cartography on the web. Within this generic word different applications are grouped covering a wide range spectrum from

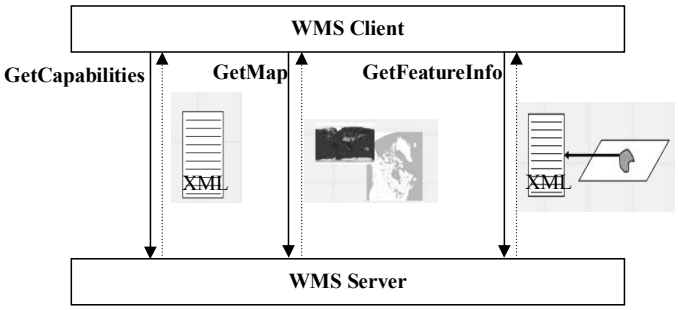

Fig. 1. WMS GetCapabilities, GetMap and GetFeature operations.

Color image of this figure appears in the color plate section at the end of the book.

a simple visualization with a web browser to a dynamic and interactive cartographic tool. The dynamic component is provided for example by pan-zoom, modulations of displayed layers or some widgets such as tooltips: e.g., Google Maps or Bing Maps. To provide such facilities, the GetFeatureInfo operation requires a set of parameters. The server can therefore build a structured result based on the database schema. The query can be reformulated by: "What is there here?" Here is defined by a couple of coordinates. The output format is an array in HTML or a XML tree.

Current results of a WMS service are generally images. The intrinsic structure of a result reduces possible manipulations. An end-user does not manipulate structured data but an image. To be able to access data, the end-user must select an object on the image. The GetFeatureInfo operation provides this selection. Nevertheless, some limits appear in the sense that the provided schema is the set (or a sub-set) of available data for this/these object(s). No link is performed with the environment of this object. Furthermore, an end-user must be aware of the database schema in order to select only a sub-set of available data.

WMS operations are graphic-result-oriented. WFS specifications want to deal with data access. In the following, we present operations based on the WFS specifications and in particular the GetFeature operation.

WFS

Information provided in this part is based on the standard documentation for WFS version 2.0.0 (OGC 2010b). WFS is an OGC specification for data access. It describes responses of a Web server to geographical data manipulation operations. These operations are based on the CRUD manipulations of geographic data based on alphanumeric/spatial constraints: creation (C), read (R), update (U) and deletion (D).

The formalism used to model data exchanges for the WFS specification is GML (Geography Markup Language). GML (OGC 2007a) is a XML dialect designed to encode, to manipulate and to exchange geographical data. Specified by the OGC, its main goal is to guarantee interoperability in the location-based data field.

The WFS specification defines five operations to send queries to a geographical data server and to get answers from it: GetCapabilities, DescribeFeatureType, GetFeature, Transaction and LockFeature. Figure 2 presents the basic operations of the WFS specification.

The most important operation, in our context, for a WFS server, is the GetFeature operation. This operation delivers data instances typed by features, identifies properties that should be delivered and provides the results of spatial and non-spatial queries.

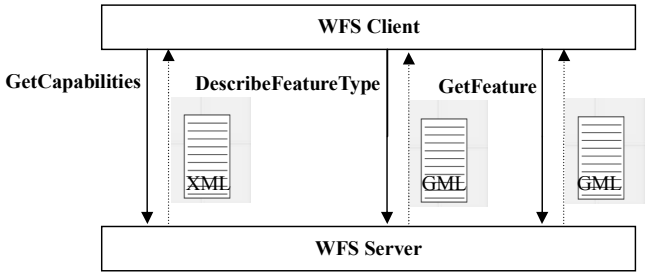

Fig. 2. Basic operations of the WFS Specification.

Selection criteria are defined using a filter. To get relevant data, a client specifies an object identifier in the filter. The WFS server receives the GetFeature query, determines the correct database, creates and sends a SQL statement to the database and formats the results. The filter is used to manage the "Where" clause of a SQL statement. Spatial data are handled with the spatial schema defined in the ISO 19107 norm. Before sending results to a client, the server transforms objects using a GML3 format.

The WFS specification emphasizes the exchange of geographical objects. Since data are distributed worldwide for a tremendous set of applications, there is no reason to define a unique and standard database schema. Therefore it is of prime importance that the data model should be provided with the data set. Unfortunately, that means that the cognitive charge of understanding the data model is directed towards end-users. In this context, we present some works devoted to the improvement of interoperability and the database schema definition.

Data-model-oriented web services

The proposition, named mdWFS, defined in Donaubauer et al. (2007a,b), defines a data-model-oriented web service. This proposition tries to take advantage of both: data interoperability as defined in OWS specifications (OGC Web Services) and the expressive power of conceptual data models. The main idea, used from conceptual data models, is the definition of a conceptual language to describe spatial data. Since the definition is provided at the conceptual level, it is independent of a specific implementation system or transfer format such as XML or GML. A wrapper (i.e., a compiler) is the ideal tool to move from a generic representation to a specific one. The semantic interoperability is provided by a representation of conceptual models. The representation of a conceptual model is translated from a schema A into a schema B. The model driven approach consists in four steps illustrated in Fig. 3 (Donaubauer et al. 2007a): specification of an application domain, specification of a conceptual schema associated with its UML meta-

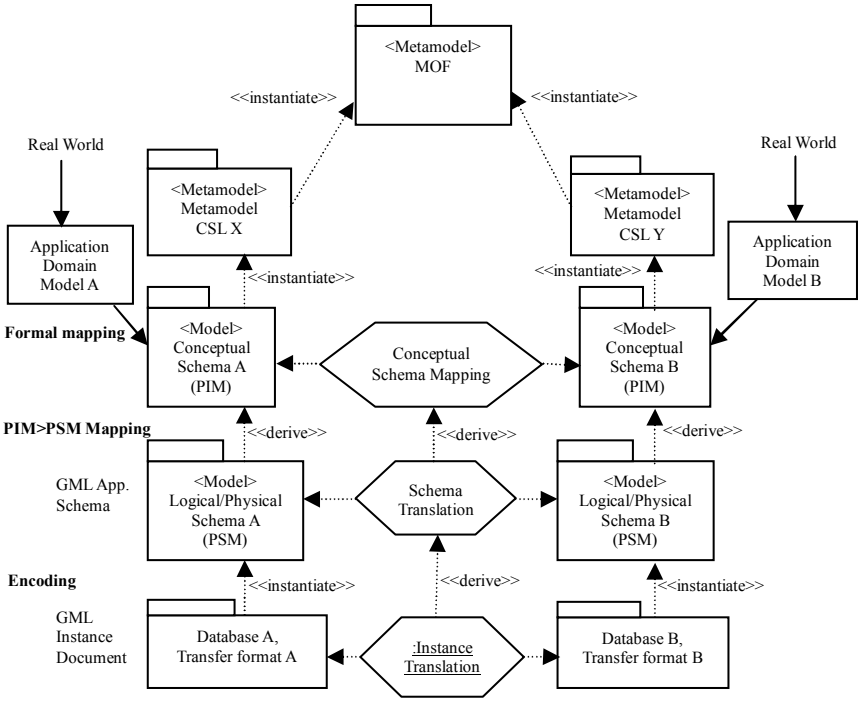

Fig. 3. Model-**driven** approach.

model, description of an application domain with a conceptual language and then derivation of the schema into various dialects.

The mdWFS proposition handles the storage / delivery of conceptual schemas and the semantic transformations depending on translation models. The first step is a semantic transformation. The second step is the configuration of a standard WFS service (OGC 2010b) to provide a data exchange service. The standard WFS service is parameterized with the destination data model but delivers data from the initial database. The WFS service must be able to store data and provide several conceptual schemas. OGC specifications should be extended.

A conceptual language, UMLT, is introduced for two main purposes: the data representation and the semantic transformations. This cartographic conceptual language is defined upon an independent extension of the UML2 meta-model. Elements of the language are specified with a UML2 model as a heritage of Activities. The textual notation of the language is defined by a set of grammatical rules (EBNF—Extended Backus Naur Form). Table 1 presents the extensions of the WFS specifications. Table 2 presents the elements of the UMLT language.

Table 1. mdWFS specifications.

Extensions	**Description**
Service (parameter)	In order to define a protocol, a new parameter in a query is defined: service = mdWFS.
GetCapabilities operation	The GetCapabilities operation is extended to obtain a SchemaList. This list contains all the available schemas in this service.
DescribeFeatureType operation	The DescribeFeatureType operation is extended to provide the XMI format (i.e., metadata exchange format in UML using XML) to transfer information on models.
DoTransform operation	A new operation. Its aim is to transfer the conceptual schema representation towards mdWFS and to set up the semantic transformations.

Table 2. Elements of the UMLT language.

Elements	**Description**
SelectionCriteria	Allows a data selection upon a logical expression.
VirtualAssociation	Allows the management of objects defined as being not linked with an association object. Imported objects may have some attributes or foreign keys (evaluated in real time) in order to provide calculated relationships.
TransformationAction	Defined as a heritage of an Opaque Action UML. It provides an element of activity that cannot be anymore structured. This is an elementary action of a transformation.
AssignmentDefinitions	Identifies expressions and primitive types as specifications.
MappingRule	Built as a composition of rule definitions, it provides cartographic objects.
AssociationBinding	Selecting associated objects, it is possible to define the way these associations are evaluated.
JoinType	An enumerate type to specify the type of join within an association link.

This new service provides a better semantic interoperability in the sense that it describes a methodology. The semantic transformations are formalized at the conceptual level. During a semantic transformation, virtual associations may be associated with, using a virtual link if necessary. The virtual association is defined to provide an explicit link for the join operation (in opposition to a conventional derived association). The joinCriteria property specifies the join conditions. The new service, mdWFS, allows a higher semantic interoperability. The relevant level is a transformation at the semantic level using a conceptual level of abstraction. The results of the data schema transformation process can be queried using a GetFeature operation. For the time, this specification is not an OGC normalized service. Foerster et al. (2010) and Wiemann et al. (2012) present different kinds of transformation based on the same principles.

User oriented schema transformation

An alternative of mdWFS service is to let the user guide transformations. Bucher and Balley (2007) define a methodology to get a transformation schema to adapt data to a specific application. The idea is, first, to transform the initial data model to adapt it to new requirements, then to get the geographic data instances. Modifications of the initial data model are performed step by step (e.g., selection, modification of the data structure) to obtain the final structure. These modifications define an algorithm to be applied to get instances from the database. This is equivalent to a workflow application illustrated in Fig. 4 (Bucher and Balley 2007), with an imperative programming language, step-by-step modifications are specified and then

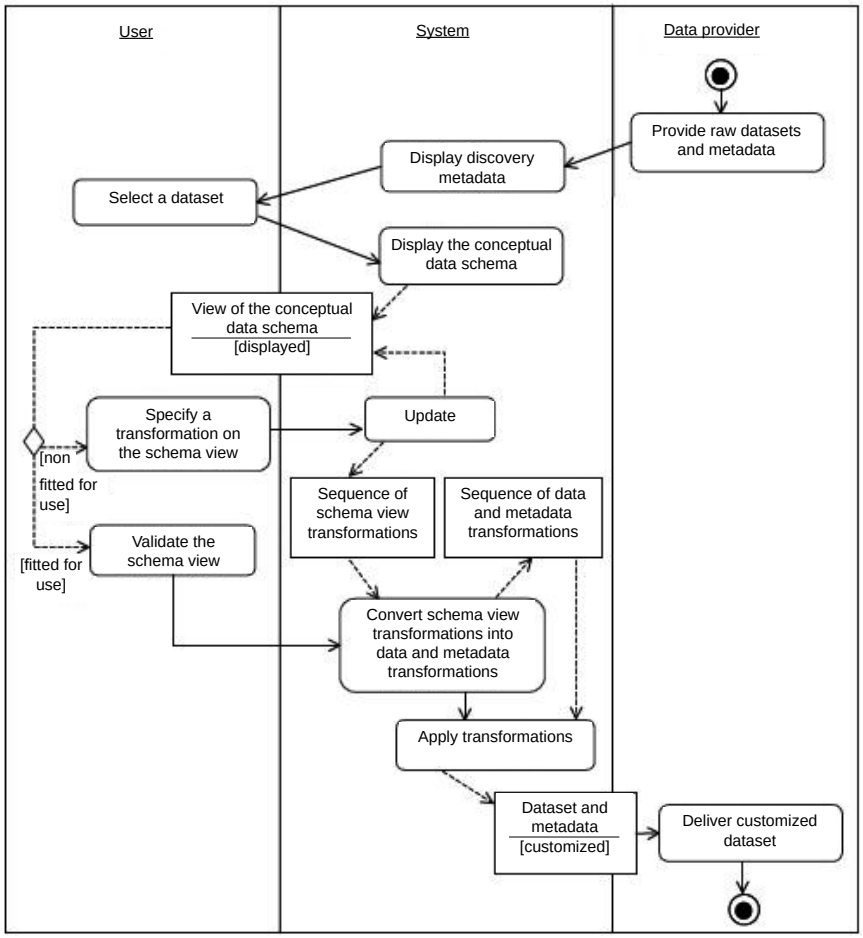

Fig. 4. Activity diagram: customization of a dataset.

applied. During a transformation, the system is in charge of the coherency of the dataset.

Proposed operations to manage the conceptual schema are applied on classes (in the sense of object oriented modeling), attributes, objects and relationships. The language allows manipulations upon classes with the definition of aggregates, renaming, deletion, specialization and generalization; upon attributes, with renaming and deletion; upon objects with aggregation and filter; upon relationships, with creation and deletion.

Some perspectives of this work are the semi-automation of the final schema, the use of web services to transform database schemas (direct or indirect—i.e., issued of a web service) or a standard manipulation to provide a predefined structure as results of the transformation.

One of the main differences between the approaches developed in (Staub 2007; Staub et al. 2008) and (Bucher and Balley 2007) is based on the transformation specification. Staub's approach relies on a mapping specification from the initial schema to the destination schema. The mapping is defined with selection and transformation operators. The user specifies transformations based on schema manipulations instead of defining an imperative approach of restructuring tasks. This approach has a better relevance whenever the destination schema is known and the transformation specification is based on a method rather than operations. The mix of these two approaches leads to promising perspectives.

Synthesis

Described propositions show weaknesses on the topic of data semantics in web services. OWS such as WMS and WFS are database schema dependent. OWS offer a syntactical interoperability (e.g., data exchange) but nothing about semantic representations (e.g., data model relationships). Conceptual data models are hidden, and semantic transformations are not available. One of the reasons may be similar to the introduction of external schemas in relational databases (i.e., view mechanism). The lack of semantic tools may be another one. We propose to reduce the second one by providing a way to introduce semantic aspects while exchanging data.

WMS services mainly provide graphic results since WFS deals with data access. The visual aspect, rather than complex functions, should be kept in mind while evaluating consultation tools based on Internet solutions. The choice is based on requirements and constraints on services: simple consultation (data and interactions with images) or real integration of data sources in complex applications (i.e., interactions with data based on the CRUD paradigm).

It is of prime importance to integrate the use of WFS operations. In the current state of OWS, a requirement based on the visualization of a huge volume of data leads to use a WMS service. This service is better fitted and widely sufficient. Furthermore, if the requirement is to access to specific data, to modify on the fly the representation of specific data or to update them then WFS operations are mandatory. Nevertheless, this service must be used with precautions since the involved volume of data may be important and therefore the response time may be significant.

The goal of our proposal is to improve OWS such as WMS and WFS, mainly the GetFeatureInfo and GetFeature operations. The aims are to provide a better relevance of alphanumeric data to improve spatial analysis and to provide a data model schema to handle such information. The second proposition we presented in this section deals with rules to determine a conceptual schema but does not consider instantiations of such models. This is the matter we want to tackle.

Data Conceptual Schema

A selection of a (geo)graphical object (e.g., city, restaurant) represented on a map, is translated into a query that will retrieve information about this object. This knowledge (schema and data) that can be considered as static, is provided by default to all users. It does not correspond in any way to users' point of view. We, therefore, suggest adapting the database schema associated with a query result, by withdrawals of irrelevant attributes and / or additions of relevant attributes. We propose a mechanism of refinement for a query. It is a way to improve the search performance of relevant attributes. It can be based, for example, on a process that adds to users' queries new attributes related to those initially present in these queries.

Provided information will be richer and more "dynamic" than with a simple use of a WFS call. Indeed, some additional attributes are the result of methods (here functions) applied, among other things, on the attributes associated with a spatial object. These methods are completely transparent for an end-user. Provided information will always be, by the end, presented as a set of attributes.

Obviously, this new database schema cannot be obtained from current distant database schemas. We are therefore interested in determining a more sophisticated database schema that integrates semantic links, attributes and methods. We stand here at the conceptual level and introduce a semantic extension that will allow the definition in a database schema, of relevant attributes and methods. This requirement is related to the impossibility of defining external schemas considering the heterogeneity of potential clients. Figure 5 illustrates our approach that consists in enriching the data schema.

Starting from an extended schema with our new concepts, we proceed in two phases. The first phase removes attributes that are considered as irrelevant with respect to a specific user's point of view. The second phase introduces new relevant attributes.

To illustrate these new extensions, we use an example of a conceptual schema formalized in UML and presented in Fig. 6, as a class diagram. It describes self-explaining geographical entities: cities, countries, hotels, restaurants, trees and green spaces. Each class is formalized by a set of attributes and a set of methods. SR stands for Spatial Representation defines on the OGC Geometry domain. Each method is defined as a function (non-function methods are not visible in the context of this chapter).

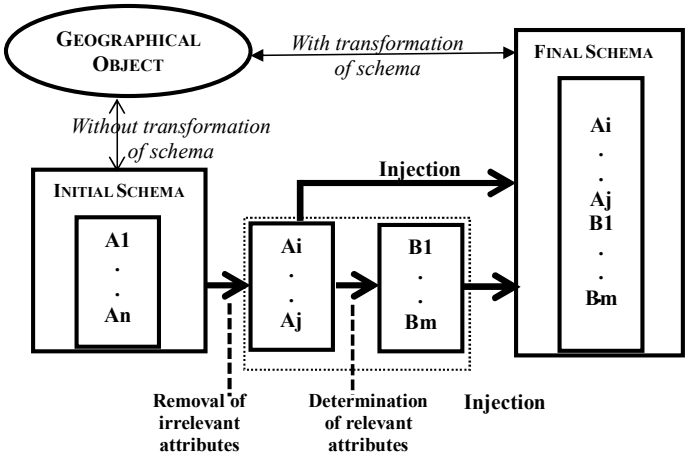

Fig. 5. Enrichment process of data schema.

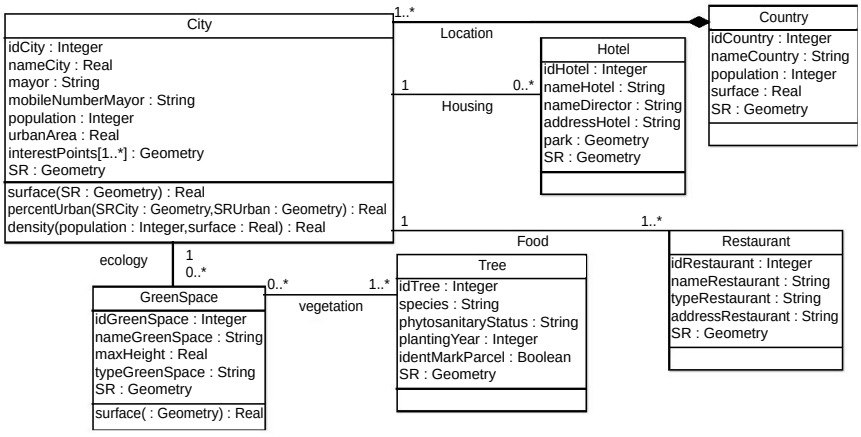

Fig. 6. Sample of a conceptual schema.

Notations: The following notations and conventions are used throughout the chapter. {} denotes the set constructor and () denotes the list constructor. Let C = $\{c_1,..., c_p\}$ be a set of classes. Let A = $\{a_1,..., a_m\}$ be a set of attributes. Let M = $\{m_1,..., m_n\}$ be a set of methods. For each class $c_i \in C$, we denote $c_i = (A_i, M_i)$ a class with $A_i \subseteq A$ and $M_i \subseteq M$. Let K_i be a set containing the attribute(s) defining a key for a given class c_i with $K_i \subseteq A_i$. The definition of a class c_i becomes $c_i = (A_i, M_i, K_i)$. The list $(p_1, .., p_q)$ denotes the parameters of a method m_k as $m_k \in M$. OP denotes the available set of spatial operators (e.g., orthometric distance noted $\surd$, inclusion noted $\subset$). FD denotes a Functional Dependency. RL denotes a Relevant Link.

This section is organized into five parts. In the first part, we present the concept of functional dependency and its weaknesses. In the following three sections, we present the three types of proposed Relevant Links. Finally, we conclude this section by giving in a fifth part a summary of our proposals.

Functional Dependencies

Functional Dependencies (FDs) are one of the key concepts of relational database schema definition.

Definition

The first semantic link introduced in the design of a relational database schema is the functional dependency. It expresses a dependency between two (sets of) attributes at the functional level. The formal definition in the relational database context is the following: X functionally determines Y if, whatever relation r is the current value for R (a relation), it is not possible that r (an instance of R) has two tuples that agree in the components for all attributes in the set X yet disagree in one or more components for attributes in the set Y (Ullman 1995). Table 3 presents a representation of Functional Dependencies in a class context.

Table 3. Formal definition of a functional dependency.

Let $c_i = (A_i, M_i, K_i)$ be a class where $c_i \in C$. Let $A_s = \{a_{s1},..., a_{sn}\}$ be a set of attributes. Let $A_c = \{a_{c1},..., a_{cm}\}$ be a set of attributes
Pre-conditions: $A_s \subset A_i$ and $A_c \subset A_i$ and $A_s \cap A_c = \varnothing$
There is a Functional Dependency between A_s et A_c noted $A_s \rightarrow A_c$ then:
$\forall$ I_1, I_2 two instances of c_i,
$(I_1.a_{s1}, ..., I_1.a_{sn}) = (I_2.a_{s1}, .., I_2.a_{sn}) \Longrightarrow (I_1.a_{c1}, .., I_1.a_{cm}) = (I_2.a_{c1}, .., I_2.a_{cm})$

As an example, there is a FD between the attribute mobileNumberMayor and the attribute mayor of the City class. This FD is written with an arrow: "mobileNumberMayor → Mayor". The semantic interpretation of such an FD denotes that a mobile phone number relates only to a single mayor (but that does not mean anything about the fact that a mayor may have several mobile phone numbers).

Weaknesses

Functional Dependencies do not express all data semantics. They do not in fact represent any link that may exist between data from the application point of view. For example, regarding the Tree class it is not possible to link the attributes "species" and "phytosanitaryStatus" with a FD. Indeed, a value of the attribute "species" does not determine a unique value of the attribute "phytosanitaryStatus" (several trees can be of the same species with different phytosanitary status). With the "species" attribute of a tree, it would be useful to obtain, from an agronomic point of view, its phytosanitary status but the identifying mark of the parcel is not relevant (for which there is no FD).

A FD does not allow establishing a link between the attribute "population" and the attribute "density". The value of density is calculated from the attribute "area" and the attribute "population" whose value is obtained using the "surface" function. Once the population is available, it would be useful, from a social point of view, to provide the density.

The independence of attributes (e.g., "Park" of Hotel and "typeGS" of GreenSpace) does not allow, for the application of a spatial operator (e.g., "inclusion" on spatial representations), obtaining information through a functional dependency. With a park, it would be useful to obtain from a tourist point of view, types of green space associated with.

The functional dependencies are not sufficient to recover, at the request of a user, the relevant attributes (single or calculated). It is important to establish a link that allows, from a database schema, obtaining a set of relevant attributes with respect to the user's point of view. This requirement is related to the impossibility of defining external schemas considering the heterogeneity of potential users.

We present in that sense, three types of links called relevant Links (RL). The first type is called RLintA-A and concerns relevant links defined between attributes of the same class (the meaning of the acronym is R(elevant)L(ink) Int(ernal)A(ttributeTo)A(ttribute)). The second type is called RLintA-M and concerns relevant links defined between an attribute and a method of the same class. Finally, the third type is called RLextA-A and concerns relevant links defined between attributes of different classes.

Relevant Link RLintA-A

A RLintA-A link leads to remove irrelevant attributes in the case where the initial schema has all the attributes associated with an object. Otherwise, it allows the determination of relevant attributes that are not included in the initial schema.

A RLintA-A link binds two sets of attributes of a class, possibly reduced to a singleton. This RL is defined by the following elements: the class concerned by the RL (c_i), the set of source attributes (A_s) and the set of target attributes (A_t). These two sets, A_s and A_t, are disjoint. We say that the source attributes semantically determine the target attributes. Table 4 presents the formal definition of RLintA-A.

This definition means that in the initial schema where it exists the set of (source) attributes $\{a_{s1},..., a_{sn}\}$, this RL adds in the final schema, the set of (target) attributes $\{a_{t1},..., a_{tm}\}$. Note here that all the attributes are not necessarily involved in this RL (e.g., key attribute). In fact, the key is not taken into account because keys operate at the functional dependency level in a database schema design.

As an example, from an agronomic point of view, a user who already has tree species, the phytosanitary status of the trees represents relevant information. Therefore, we define a RLintA-A link between two attributes of the Tree class (e.g., "species" and "phytosanitaryStatus"). This RL is defined by: -A_A> (Tree, species) = phytosanitaryStatus.

Operational complexity depends on the implementation of the database. The database administrator is in charge of defining such a complexity. It may be constant as defined for example in a framework of a relational database with sets materialization of source and target attributes in the same relation (the same tuple). It can be higher due to the third normal form, with or without index(es) in the second (third) relations. The complexity may be changed out of computed attributes (i.e., the complexity depends on the complexity of the evaluation).

Table 4. Formal definition of RLintA-A.

Let $c_i = (A_i, M_i, K_i)$ be a class where $c_i \in C$. Let $A_s = \{a_{s1},..., a_{sn}\}$ be a set of attributes. Let $A_t = \{a_{t1},..., a_{tm}\}$ be a set of attributes
Pre-conditions: $A_s \subseteq A_i$, $A_t \subseteq A_i$, $A_s \cap A_t = \emptyset$, $K_i \not\subset A_s$, $K_i \not\subset A_t$
A RL of **RLintA-A (-A_A->)** type between A_s and A_t is defined by:
-A_A-> : C x A x .. x A -> A x .. x A
-A_A-> $(c_i, a_{s1},..., a_{sn}) = (a_{t1},..., a_{tm})$

Relevant Link RLintA-M

A RLintA-M link leads to additional attributes in the final schema, which are results of methods.

A RLintA-M link binds a set of attributes of a given class (possibly reduced to a singleton) to a method of the same class. This set of attributes represents the source attributes and belongs to the initial schema. The result of the method (here a function) will be added to the final schema as a new attribute.

This RL is defined by the following elements: the class concerned by this RL (c_i), the set of source attributes (A_s), the method (m_k), its parameter list (P). Table 5 presents the formal definition of RLintA-M.

This definition means that in the initial schema where it exists the set of attributes $\{a_1,..., a_n\}$, this RL adds in the final schema an attribute that denotes the result of the method m_k ($a_{|A|+1}$). Note here that several links can possibly call the method m_k.

The recursive definition of the set A, with respect to the creation of attributes denoting the link, leads to establish (recursively or not) links between methods. One of these methods is considered as a source attribute. The origin of the recursion is necessarily, by the definition of the link, in the set of non-calculated attributes.

As an example, from a social point of view, we illustrate this type of RL between the attribute "population" and the method "density (integer, real)" of the City class. This RL is defined by: A_M-> (City, population, density

Table 5. Formal definition of RLintA-M.

Let $c_i = (A_i, M_i, K_i)$ be a class where $c_i \in C$. Let $A_s = \{a_{s1}, ... , a_{sn}\}$ be a set of attributes . Let $m_k \in M_i$ be a method. Let $\|A\|$ denote the cardinality of the set of attributes A. Let $a_{\|A\|+1}$denote the result attribute. Let D be a domain. Let P be defined as P: D P | M (P) P | ε
Pre-conditions: $A_s \subseteq A_i$, $K_i \not\subset A_s$
A RL of **RLintA-M (-A_M->)** type between A_s and $a_{\|A\|+1}$ is defined by:
-A_M->: C x A x … x A x M x (P) -> A $(c_i, a_1, \dots , a_n, m_k, (p_1, ... , p_q)) = a_{\|A\|+1}$
Post-condition: $A = A \cup \{a_{\|A\|+1}\}$

(population, surface (SR))) = density. The source attribute "population" semantically determines the method "density (integer, real)". This method has the following parameters: the attribute "population" and the result of the method "surface (SR)". The target attribute, here called "density", is provided as a new attribute in the final schema as a conventional attribute. It can thus be directly provided in complement of data schema information given by the WMS / WFS operations GetFeature or GetFeatureInfo.

Operational complexity concerns the complexity of the applied methods. In the case where a method has parameters that are results of methods, operational complexity will be aligned with the complexity of these methods.

We emphasize here that there are two approaches to the determination of methods. The first approach is to use external services. WPS (OGC 2007b), defines the meaning of rules to set up and to run a geo-processing as a Web service. The user executes in this case a geospatial operation, with all necessary data. The server starts the process and informs the user of its progress. An architectural alternative to this approach is given in Grosso et al. (2009) or Stollberg and Zipf (2007) with an orchestration of services. The second approach is internal and is linked to the database. Our proposal is oriented towards the second approach. These two approaches are complementary in the sense that the method could be the implementation of a Web service request (e.g., a WPS). The Web service is an encapsulation of the method and the method or the WPS are hidden to final users.

Relevant Link RLextA-A

A RLextA-A links leads to additional attributes in the final schema, which are retrieved from one or more classes.

A RLextA-A link binds two sets of attributes (possibly reduced to a singleton) of different classes. The classes involved in these links may be of relevance related among others, through the inheritance relationship, composition or be completely independent in the conceptual data schema. This RL is defined with a spatial operator in order to make an independent connection between different classes of the data model. It therefore represents a semantic spatial link.

RLextA-A is defined by the following elements: the class concerned by the RL (named of source class), the set of source class attributes, the target class, the relevant target class attributes, the spatial query involving the source class and target class. Table 6 presents the formal definition of RLextA-A.

This definition means that in a schema where exists the set of attributes $\{a_{s1},..., a_{sn}\}$ of a class c_i, this RL adds in the final schema the set of (target) attributes $\{a_{t1},..., a_{tm}\}$ of the class c_j.

Table 6. Formal definition of RLextA-A.

Let $c_i = (A_s, M_i, K_i)$ be a class where $c_i \in C$. Let $c_j = (A_j, M_j, K_j)$ be a class where $c_j \in C$. Let $A_s = \{a_{s1}, .., a_{sn}\}$ be a set of attributes. Let $A_t = \{a_{t1}, .., a_{tm}\}$ be a set of attributes. Let REQSPAT defined as REQSPAT: OP (ReqSpatP, ReqSpatP)
ReqSpatP : Geometry | OP (ReqSpatP ReqSpatF)
ReqSpatF: ',' ReqSpatP | ε
Pre-conditions: $A_s \subseteq A_i$ and $A_t \subseteq A_j$ and $K_i \not\subset A_s$
A RL of **RLextA-A (e-A_A->)** type between A_s and A_t is defined by:
e-A_A-> : C x REQSPAT x A x .. x A -> C x A x .. x A
(c_i, reqspat, a_{s1}, .., a_{sn}) = (c_j, a_{t1}, .., a_{tm})

PI check

Example 1 (1.~~:~~ linked classes): From a tourist point of view, a user may want to eat in a restaurant near a point of interest. It would be appropriate to determine the types (e.g., international, French, Italian) and addresses of Restaurants within a reasonable distance. The administrator defines a RL between the spatial attribute "interestPoints" of the City class and alphanumeric attributes "typeRestaurant" and "addressRestaurant" of the Restaurant class. This RL is defined by: e-A_A-> (City, reqspat, interestPoints) = (Restaurant, typeRestaurant, addressRestaurant), where reqspat is a spatial query. This query is defined by the application of the √ operator: √(interestPoints, SR < 200) on the spatial attributes "interestPoints" of the City class and "SR" of the Restaurant class.

Example 2 (independent classes): From a tourist point of view, a hotel may have a park located in a wider green area. It would be appropriate to determine the name and type of this green space. The administrator defines a RL between the spatial attribute "park" of the Hotel class and the alphanumeric attributes "nameGS" and "typeGS" of the GeenSpace class. This RL is defined by: e-A_A-> (Hotel, reqspat, park) = (GreenSpace, nameGS, typeGS), where reqspat is a spatial query. This query is defined by the application of the ⊂ operator: ⊂(park, SR) on spatial attributes Park of the Hotel class and SR of the GreenSpace class.

Operational complexity is defined with two stages. The first stage concerns the complexity of the spatial operator. In a second stage, this complexity can be modified by the research of the potential instance(s) for

this link. A value depends on the physical implementation of the database. For example, a specialization can be translated into a set of relational joins or a set of projections and unions.

We emphasize that the goal here is not to propose a standard solution like the OGC Table Joining Service (TJS) (OGC 2010a). TJS offers a method of data research, access and use from multiple sources and, dynamically, to supply the databases, to perform analysis. Indeed, we use the same schema of database to determine the relevant attributes.

Synthesis

The concept of relevant link (RL) presented in this section widely extends the functional dependence. They take into account more semantics such as the ones that may exist between attributes and between attributes and methods. The main advantage of this proposal lies on semantic definitions. Information will be processed with links of relevance and not only from their graphical representations.

Figure 7 presents the relationships between the various concepts of our meta-model formalized here in UML.

This diagram indicates the following rules:

- A class can have the role of source class and / or of target class within a RL.
- A class is composed of attributes, methods, and possibly RL.
- An attribute can be spatial or alphanumeric.
- An attribute can have the role of a source attribute or of a target attribute within a RL.
- An attribute has a single type.
- A method has a list of parameters and a return value defined by a type.
- A parameter can be an attribute or the result of a method.
- A constraint (pre-conditions and / or post-conditions) may be associated with an RL.
- A RL can be typed by RLintA-A, RLintA-M or RLextA-A.
- A RLintA-A establishes a link between two sets of attributes of a class.
- A RLintA-M is built with a set of attributes and methods of a class, the parameter list of this method and its result. A typed attribute denotes the result.
- A RLextA-A can bind two sets of attributes of two different classes.
- A RLextA-A does or does not involve a spatial query.

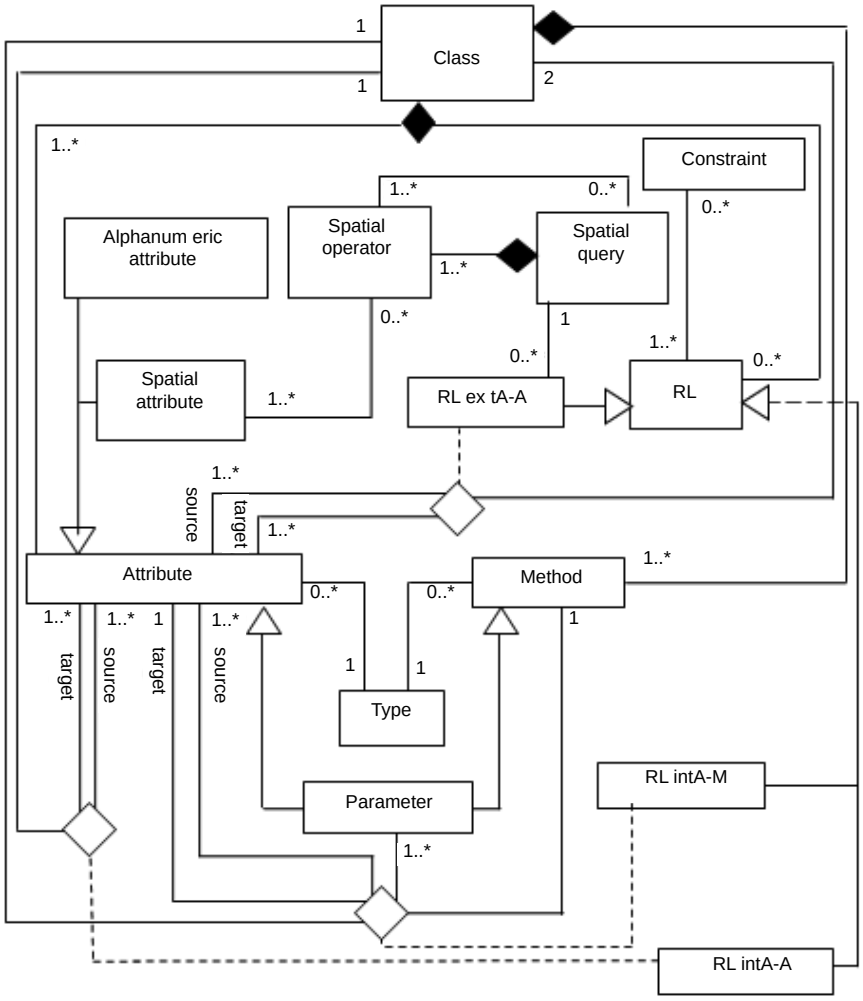

Fig. 7. Meta-model formalized in UML.

- A spatial query is composed of one or more spatial operators.
- An operand of a spatial operator is a spatial attribute or a spatial query.

In the following, we present the classification and the determination of relevant RL for GetFeatureInfo and GetFeature operations. These operations will provide relevant attributes from the database schema.

Determination of Relevant Links

At this stage, the problem is not the connection with the Data Base Management System but the identification and the choice of relevant attributes among the database schema. They should satisfy the better relevance criteria. The identification is therefore equivalent to determine the Relevant Links (RL). In that sense, we propose to classify the RL in order to facilitate the search.

Several formalisms exist to handle a classification, such as graph-based or logic-based paradigms for example (Bedel et al. 2007). We propose to use a lattice of concepts for the classification of RL. Foerster et al. (2010) and Messai et al. (2008) modify the lattice structure while analyzing end-users' query. In contrast to this approach, we propose starting from the data model of GetFeature or GetFeatureInfo data model to add or to delete attributes in order to define the data model associated with end-users' queries. This approach is similar to a conventional viewed mechanism in relational database.

Conventional applications in Information Systems have the knowledge of the database schema (eventually with the formalism of views). In our context, applications use a Web service that accesses to a distant database. Users do not have an a-priori knowledge of the database schema nor the database administrators have an a-priori knowledge of end-users' needs. Therefore a view mechanism cannot be provided. Nevertheless, data base administrators have a very good knowledge of the database schemas that are offered to external queries.

We proposed the concept of Relevant Links that provides a semantic relationship between database schemas and users. Starting with a user query, we propose to offer a semi-automatic design of a result database schema. The Data Base Administrator is no longer in charge of providing static external schemas (i.e., views in relational databases) for the set of queries. A middleware on the clients' computers is in charge of providing a dynamic "view". This operation is transparent to end-users. The "view" is relevant to an end-user's point of interest and is designed according to the operational complexity (s)he is willing to accept.

This section is organized into four parts. In the first part, we present the formal definition of lattice of concepts. In the second part, we present the links of this lattice and the available RL. In the third part, we present the extended formal definition of a RL. Then, in the fourth part, we present the determination of RL.

Lattice Formalization

We propose a classification built over a partial ordered structure named lattice of concepts (Wille 1982; Nauer and Toussaint 2009). This structure

is induced by an environment (context) E, has a set of properties, T, a set of structures, L, and a binary relationship, I, named incidence of E, such as $I \subseteq T \times L$.

Let $L' \subseteq L$, be a set of structures. The set of common properties to the structures of L' is $X = \{x \in T \mid \forall y \in L', (x, y) \in I\}$.

Let $T' \subseteq T$, be a set of properties. The set of structures with all the properties in T' is $Y = \{y \in L \mid \forall x \in T', (x,y) \in I\}$.

A concept is formally represented with a couple (X, Y). Let F be the set of the concepts, relevant to an environment E. Let f_1 be (X_1, Y_1) and f_2 be (X_2, Y_2) in F, f_1 is subsumed by f_2 (noted $f_1 \leq f_2$) if $X_1 \subseteq X_2$ or from the dual point of view $Y_2 \subseteq Y_1$. $(F, \leq)$ is a complete lattice, named lattice of concepts linked to the formal environment E.

The lattice of concept we propose to use, references to a classification tool of the RL as an ontology of domain. This lattice describes concepts involved in a specific domain in order to take into account the different links between the components of this domain. Numerous works deal with the modeling (in the ontology sense) such as (OGC 2008) with CityGML. CityGML, recognized as a norm, provides a classification for 3D modeling of a town. Emgard and Zlatanova (2008) propose a generic meta-model for a 3D representation as a complement to CityGML.

In comparison with these works, we propose a meta-base (instead of a meta-model) with the formal representation of elements, rules and constraints that regulate the creation of a database (but not the application domain).

Within the lattice, a concept represents, from a semantic point of view, the level of interpretation for a group of data. This level is relevant for a specific user.

Let E = (T, L, I) be a formal context. T is a set of themes. L is a set of Relevant Link (RL). I is an incident function of E such as $I \subseteq T \times L$ and $(t, l) \in I$, $t \in T$ and $l \in L$. That means that theme "t" is a property of the Relevant Link "l".

Let t be a theme such as $t \in T$. The set of RL sharing this theme is L' such as $L' = \{l \in L \ / \ (t, l) \in I\}$.

A formal concept is a theme shared by a set of RL. It is formally represented by a couple (t, L') where $t \in T$ and $L' \subseteq L$.

Extended Definition of Relevant Links

To provide a classification of Relevant Links (RL), we propose to extend the definition of a RL (given in section 3). Let N denote the set of natural integers. The extensions are: an identifier (i.e., id, such as $id \in N$), a theme (i.e., $t \in T$) and a complexity θ associated with the process in charge of obtaining data. This complexity is data base implementation dependent.

Notation: Let n be a variable defining the quantity of data (e.g., the number of tuples in a database). Let T be a set of themes and θ = {O(1), O(log(n)), O(n), O(n^2), O(>)} a set of complexity (Papadimitriou 1993).

The formal definition of RL typed by RLintA-A becomes:

-A_A->: N x T x θ x C x A x … x A → A x … x A

The formal definition of RL typed by RLintA-M becomes:

-A_M-> : N x T x θ x C x A x … x A x M x (P) → A

(id, t, o, c_i, a_1, … a_u, m_k, (p_1, …, p_q)) = $a_{|A|+1}$

The formal definition of RL typed by RLintA-A becomes:

e-A_A-> : N x T x θ x C x REQ-SPAT x A x … x A → C x A x … x A

(id, t, o, c_i, request-spatial, a_1, … a_u) = (c_j, a_v, …, a_m)

Application of a lattice to Relevant Links

We propose to use a lattice of concepts, named here, lattice of reference, to classify Relevant Links. The Data Base Administrator is in charge of defining a theme to a Relevant Link using this lattice. Figure 8 presents an example of a (reference) lattice of concepts in our example. This lattice is shared by the set of applications that wish to use the enhancement service (in order to keep unchanged the signature of the normalized web services).

To illustrate the formal context E, we rely on a set of RL and themes presented in Table 7. Symbols are used to provide a better reading. An example of a formal context is presented Table 8.

Figure 9 presents the associated lattice of concept for a formal context E. This lattice is named Instantiated Lattice of Concepts. It corresponds to a sub set of the lattice of concepts and is specific to a geographical data server.

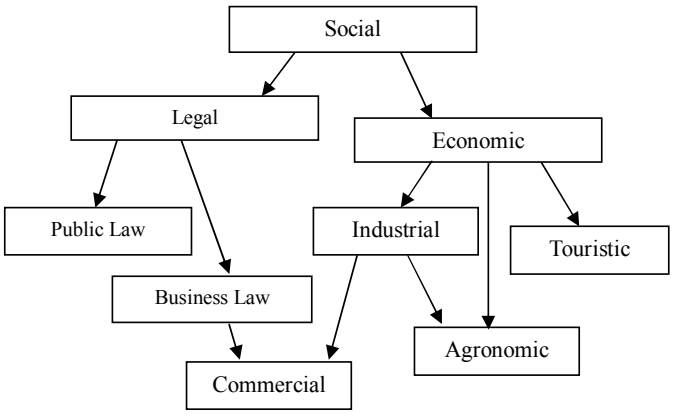

Fig. 8. Example of reference lattice of concepts.

Table 7. Designation of RL.

Designation of RL	Symbol
-A_A->(Tree, species) = phytosanitaryStatus	l1
-A_M->(City, population, density, (int, surface(RS))) = density	l2
e-A_A->(City, √, interestPoints) = (Restaurant, typeRest, adressRest)	l3
e-A_A->(Hotel, ⊂, park) = (GreenSpace, nameGS, typeGS)	l4
-A_A->(City, urbainArea) = mayor	l5
-A_A->(Country, population) = pib	l6
-A_M->(City, urbainArea, percentUrban, (urbainArea, surface(RS))) = percentUrban	l7

Table 8. Example of formal context E.

	l1	l2	l3	l4	l5	l6	l7
Social	0	true	0	0	0	true	0
Legal	0	0	0	0	true	0	0
Economic	0	0	0	0	0	0	true
Touristic	0	0	true	true	0	0	0
Agronomic	true	0	0	0	0	0	0

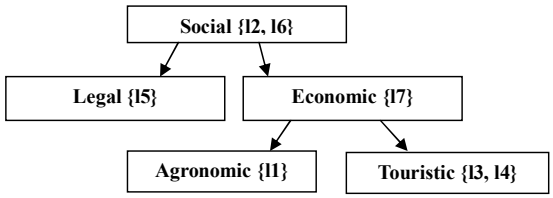

Fig. 9. Instantiated lattice of concepts.

Determination of Relevant Links

We consider in this section that a user query is based on two components: a temporal component (i.e., service time) and a semantic one.

About the temporal component, the user is invited to define the acceptable level. (S)he can accept the standard data provided by WMS and WFS services or can request for enhanced data. (S)he first indicates the maximum acceptable complexity to get these data (e.g., logarithmic, linear, quadratic). The complexity is widely linked to the database structure on which distant end users cannot interact. Her / His level of acceptation depends on her / his center of interests and may vary during the spatial analysis process. The temporal component is defined for each theme and provides an Operational Lattice of Concepts. This lattice is a sub-set of an Instantiated Lattice of Concepts and contains RL that are compatible with the temporal complexity for this theme. In the event of a change during

the spatial analysis process, a new Operational Lattice of Concepts is generated.

The semantic component represents the position in the lattice. The user specifies her / his semantic requirement: asking enhanced data for a specific theme or indicating a theme and applying a closure operator on the graph modeling the lattice. This allows, for each query, the localization of the treatment(s) associated with the dynamic construction of a view into the end-users' side. This construction is completely transparent to geographic data servers.

In order to simplify the presentation, let us consider that the Instantiated Lattice of Concepts (Fig. 7) also corresponds to the Operational Lattice of Concepts. The application of the GetFeatureInfo or GetFeature operations will be applied to an instance (of a map or an object in a given map). The problem is now transformed into the definition of the data model associated with this operation.

The GetFeatureInfo operation (from a WMS service) will require the traversal of links typed by RLintA-M and by RLextA-A in order to define relevant attributes to be added to the initial schema. The standard GetFeatureInfo operation provides the whole schema associated with the result. The RLintA-A typed links are useful to determine attribute(s) to be taken off from the initial schema. To take into account the typed links RLintA-M and RLextA-A, the definition of a theme is essential. Based on the RLintA-M link, the definition of the theme "Economic" will lead to only traverse the link {"L7"}. The definition of the theme "Economic" with recursive application will lead to traverse the links {"L7"} * {"L3", "L4"}. Based on the RLintA-A link, the definition of the theme "Economy" will lead to only traverse the link {"L6"}. The source and destination attributes involved in this link and the source attributes of the links typed RLintA-M and RLextA-A will be considered as relevant and will be kept in the final schema. All other attributes will be removed (except the key attribute(s) if this last one is not already present in the final schema).

Table 9 illustrates an example of interactions of RL in a query based on the class "Town" with the theme "Economic".

To deal with the GetFeature operation (from a WFS service), we provide two approaches. In the first approach, we suggest automatically generating a schema from the user query and to limit the result to the theme and the key attributes. In this case, the GetFeature operation will lead to the traversal of RLintA-A, RLintA-M or RLextA-A typed links. Within the same definition of a theme (e.g., "Economic"), links {"L6", "L7"} will be traversed. The final schema will be composed of the different attributes involved in the traversed links. In a second approach, named standard, the user selects attributes that should be involved in the final schema. The GetFeature operation will require the traversal of links typed "RLintA-A". This traversal is performed

Table 9. Example of GetFeatureInfo query.

Id Town	urbainArea	percentUrban	interestPoints	TypeRest	Address Rest

Economic Touristic

if the source belongs to the required attributes and at least an attribute of the destination does not belong to the required final attributes in order to define attributes to be added to the final schema. Within the same definition of a theme (e.g., "Economic") other links {"L6", "L7"} will be traversed under the same conditions for the source attributes. The process will be similar to a GetFeatureInfo operation in the case of a closure from a given theme. Let us mention that for the GetFeature operation, the schema contains relevant attributes.

Implementation

This proposition has been implemented with open source tools. Figure 10 sums up technical choices. They are compatible with norms and open standards that allow interoperability between different systems (applications, services and clients). PostgreSQL is used with its extensions: PostGIS and PgRouting for graph manipulations. Geoserver is used as a cartographical server since it provides raster and vector supports. Several databases can be handled with this server.

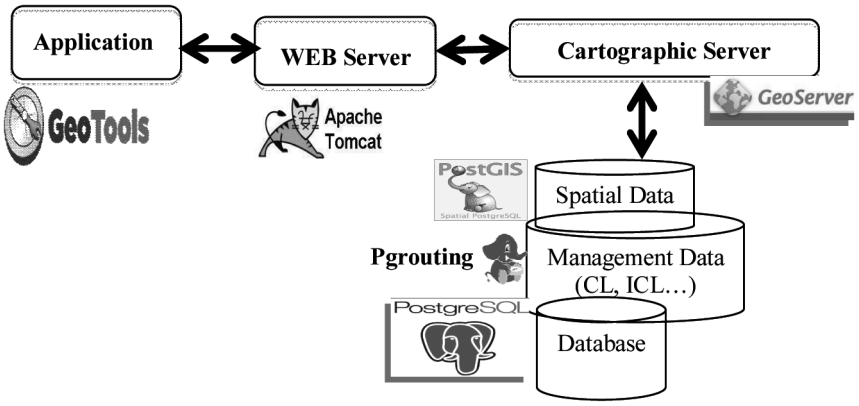

Fig. 10. Technical tools.

Color image of this figure appears in the color plate section at the end of the book.

Easily integrated in the Java Eclipse platform, Geotools is the basic component of some GIS tools (i.e., UDig). Its modular architecture and its use to develop Geoserver, leads us to use it as a basic component for cartographic applications.

From the database server side, interfaces to model RL have been realized. From the application server, using a database schema provided by a distant server, an application has been developed to enhance attribute information and to display them. A presentation of these interactions is available in (El Ghouasli et al. 2012).

Figure 11 presents the specific application interactions. An application interacts with a middleware. The first step is the connexion that will provide the Instantiated Lattice of Concepts (ICL) from the server side using WFS services. The second step is a dynamic step that adapts the received ICL to users' needs. The Operational Lattice of Concepts (OLC) is now available. Since this step is dynamic, it can be performed several times within the same working session. The third step is an access demand. Using the RL, the received (G / X)ML files from WMS / WFS services are extended or reduced depending on users' needs. The application handles these (G / X)ML files as if they were original ones provided by WMS / WFS calls.

The lattice of concepts is shared by the different applications. The Operational Lattice is specific to each database server depending on available data. The modeling of each lattice is similar since they cover the same modeling (here an acyclic graph). A conventional database can take charge of this modeling.

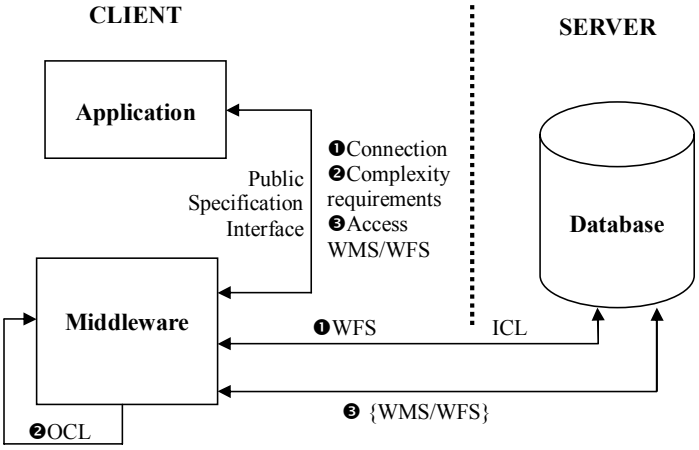

Fig. 11. Specific applications interactions.

Conclusion

To facilitate data and service interoperability, the Open Geospatial Consortium (OGC) proposes web services such as Web Map Service (WMS) and the Web Feature Service (WFS). These services allow a given system to access data from distributed data servers as soon as they both respect a common interface. These interfaces are based on operations such as for example GetCapabilities, GetFeatureInfo, and GetFeature.

These services present some weaknesses for the GetFeatureInfo operation of WMS and GetFeature operation of WFS from the semantical point of view. The DataBase Administrator of a distributed service does not know, by definition, the clients' needs (here end-users or applications using these services). The clients do not know the database schema (respectively, logical and external). Therefore, it is widely desirable to offer a mechanism close to the notion of a dynamic view in order to provide a better relevance to users' needs. This mechanism requires an enhancement of the database conceptual model from the server side.

The proposed extension defines Relevant Links (RL) between two sets of attributes belonging to the same class, between attributes and methods (i.e., functions) of the same class and finally between two sets of attributes belonging to two different classes (linked or not by constructors of the conceptual data model). Methods are viewed as attributes. In fact, users ignore the nature of available data (simple or computed). The goal is to provide a data model associated with the result of a query designed upon users' needs. The proposition must not modify the specifications of OGC norms for web services.

Relevant links, associated with classes, give a semi-automatic enhancement of the database schema provided with a result of a WMS or WFS call for GetFeatureInfo or GetFeature operations. A classification, based on a lattice, automatically gives the better enhancement of the external database schema provided with a result.

Perspectives of this work is to provide a metric evaluation of performances with real size applications focusing on the database server aspect without taking into account the network parameters (e.g., bandwidth) that are uncontrollable and not relevant for this proposal.

References

ANSI-SPARC Architecture. http://en.wikipedia.org/wiki/ANSI-SPARC_Architecture.

Bedel, O., S. Ferré, O. Ridoux and E. Quesseveur. 2007. GEOLIS: a logical information system for geographical data. Int. J. of Geomatics and Spatial Analysis, Hermes, France. 17: 371–390.

Brodie, M.L. and J.W. Schmidt. 1982. Final report of the ANSI/X3/SPARC, DBS-SG relational database task group, SIGMOD Record, Volume 12, Issue 4.

Bucher, B. and S. Balley. 2007. A generic prepocessing service for more usable geographical data processing services. http://www.agile-online.org/ Proc. of the 10th AGILE Conf. on Geographic Information Science, Aalborg, Denmark.

Donaubauer, A., A. Fichtinger, M. Schilcher and F. Straub. 2007a. Model driven approach for accessing distributed spatial data using web services—Demonstrated for cross-border GIS applications. Proc. 23rd Int. FIG congress, 2006, Munich, Germany. http://fig.net/pub/fig2006/techprog.htm.

Donaubauer, A., F. Straub and M. Schilcher. 2007b. mdWFS: a concept of web-enabling semantic transformation. Proc. of the 10th AGILE Conf. on Geographic Information Science, Aalborg, Denmark. http://agile.gis.geo.tu-dresden.de/web/index.php/conference/proceedings/proceedings-2007.

El Ghouasli, M., K. Jabri, N. Souissi and M. Mainguenaud. 2012. Improvement of cartographic information for the WMS and WFS Web services. 3rd International Conference on Multimedia Computing and Systems (ICMCS'12), May 10–12, Tangier, Morocco.

Emgård, K.L. and S. Zlatanova. 2008. Design of an integrated 3D information model. pp. 143–156. In: Coors, Rumor, Fendel and Zlatanova (eds.). Urban and regional data management: UDMS annual 2007. Taylor & Francis Group, London, UK.

Initials missing

Foerster, T., L. Lehto, T. Sarjakoski, T.L. Sarjakoski and J. Stoter. 2010. Map generalization and schema transformation of geospatial data combined in a Web Service context. Computers Environment and Urban Systems, Elsevier. 34(1): 79–88.

Grosso, E., A. Bouju and S. Mustière. 2009. Data integration GeoService: a first proposed approach using historical geographic data. J.D. Carswell et al. (eds.): W2GIS 2009, LNCS 5886, 103–119. Springer-Verlag Berlin Heidelberg.

Messai, N., M.D. Devignes, A. Napoli and M. Smaïl-Tabbone. 2008. Many-valued concept lattices for conceptual clustering and information retrieval, 18th European Conference on Artificial Intelligence (ECAI), IOS Press, Frontiers in Artificial Inteligent and Applications. 127–131M. Gallab et al. (eds.).

Nauer, E. and Y. Toussaint. 2009. CreChainDo, an iterative and interactive Web information retrieval system based on lattices. Int. J. of General System. (38–4) 363–378. Taylor and Francis (ed).

OGC, web. 2005. OpenGIS web services architecture description, Open Geospatial Consortium. http://www.opengeospatial.org/standards/bp.

OGC, web. 2006. OpenGIS Web Map Service (WMS) implementation Specification, Open Geospatial Consortium. http://www.opengeospatial.org/standards/wms.

OGC, web. 2007a. OpenGIS Geography Markup Language (GML) encoding standard, Open Geospatial Consortium. http://www.opengeospatial.org/standards/gml.

OGC, web. 2007b. OpenGIS Web Processing Service (WPS) implementation Specification, Open Geospatial Consortium. http://www.opengeospatial.org/standards/wps.

OGC, web. 2008. OpenGIS City Geography Markup Language (CityGML) encoding standard, Open Geospatial Consortium. http://www.opengeospatial.org/standards/citygml.

OGC, web. 2010a. Georeferenced Table Joining Service (TJS) implementation standard, Open Geospatial Consortium. http://www.opengeospatial.org/standards/tjs, 2010.

OGC, web. 2010b. OpenGIS Web Feature Service (WFS) implementation specification, Open Geospatial Consortium. http://www.opengeospatial.org/standards/wfs.

Papadimitriou, C. 1993. Computational complexity. Addison-Wesley.

Staub, P. 2007. A model-driven web geature service for enhanced semantic interoperability. OSGeo J. (3): 38–43. http://journal.osgeo.org/index.php/journal/issue/view/28.

Staub, P., H.R. Gnaegi and A. Morf. 2008. Semantic interoperability through the definition of conceptual model transformations. Transactions in GIS. 12–2: 193–207.

Stollberg, B. and A. Zipf. 2007. OGC Web processing service interface for web service orchestration aggregating geo-processing services in a bomb threat scenario. J.M. Ware and G.E. Taylor (eds.): W2GIS, LNCS n°4857, 239–251. Springer-Verlag Berlin Heidelberg.

Ullman, J.D. 1995. Principles of Database and Knowledge Base Systems—classical database systems, Computer Science Press.

Wiemann, S., L. Bernard, P. Wojda, P. Milenov, V. Sagris and W. Devos. 2012. Web services for spatial data exchange, schema transformation and validation as a prototypical implementation for the LPIS quality assurance. Int. J. of Spatial Data Infrastructures Research. (7): 66–87.

Wille, R. 1982. Restructuring lattice theory: an approach based on hierarchies of concepts. Formal Concept Analysis, LNCS n°5548, 314–339. Springer.