Formalism to describe multi-leveled ontologies for urban applications

El Hassan Abdelwahed
Département d'informatique, Faculté des Sciences Semlalia Marrakech
BP 23 90, Bd My Abdellah Marrakech Maroc
Tel: 212 44 43 46 49, Fax: 212 44 43 67 69
Email: abdelwahed@ucam.ac.ma

Abstract

In many applications, ontologies are used to explicit formalization of the conceptualization of the domain, but a given domain can be conceptualized through many different ontologies. On the other hand, within the last years, many ontologies are created and are accessible over the web. Thus multiples ontologies connected by semantic relations emerge as a core question and became an essential issue of interoperability between distributed applications over the web.

In this paper, we present a formalism for describing ontologies according to different levels of abstraction: functional level contains generic ontologies and the domain level contains domain ontologies. The latter are described from the former by using a derivation process. The proposed formalism is based on an algebraic approach and it allows to capture the relations over different described ontologies. To illustrate the proposed formalism, we consider an examples from the urban domain.

1. Introduction

In the early 90's , the ontologies are originally developed in the knowledge engineering community to support the knowledge sharing and reuse. An ontology provides a set of best founded constructs that can be used to build meaningful higher level knowledge. [7] defines an ontology as an explicit specification of conceptualization. In fact, an ontology is the theory about how a given domain is structured, what sorts of concepts it contains, what sorts of relationships exist among them, and so on.

With the emergence of the semantic web, the ontology research has been accelerated and advocated as support of interoperability between distributed applications over the web. The main purpose of an ontology is to support exchange and communication, without misunderstanding, among users and agents. Nowadays, ontology is being put to use for many practical purposes in many other areas. For more detailed descriptions and bibliography of the field see the published survey: [1] gives a review of ontology-based approaches to semantic integration, [2] provides a survey of the most relevant ontological modeling approaches and [3, 8] describes the current state of the art of ontological engineering and it covers theory, tools and applications.

1.1. Motivation

Different autonomously developed applications can meaningfully collaborate and communicate by using a shared ontology. unfortunately, an unique shared ontology describing all possible domains is very fast to develop and unachieved in practice. In addition, the same domain can be described by different ontologies in a heterogeneous way: the very same concept can be described in different manner and at a different level of detail by more than one ontology. On the other hand, within the last years, many ontologies are created and are accessible over the web. Semantic interoperability , therefore, can be solved by relating different ontologies via semantic mapping [4].

Guarino classifies ontologies by considering two criteria : level of detail and level of dependence. Several types of ontologies can be distinguish. In particular, there are the top-level ontologies and the domain ontologies [5]. In [6] the authors propose a multi-layered spatial ontologies definition framework for urban applications.

1.2. Contribution and organization of the paper

In this paper, we present formalism for describing the multi-leveled ontologies and their relations. To illustrate

this formalism, we consider an examples from the urban domain. In the proposed formalism, different ontologies are described according different levels of abstraction. The functional level contains generic ontologies and the domain level contains domain ontologies. The derivation process permits to describe ontologies of the domain level from those of the functional level (see Figure 1). This aspect of defining ontologies by level allows to have minimal ontologies by level. It avoids defining a global and maximal ontology which is difficult to specify completely for a given domain. The proposed formalism is based on an algebraic approach. Entities of the functional level are represented by *abstract classes*. Each class is described by its intension which is constituted by a set of its attributes and functions. The attributes and functions arguments take values in abstract data types (ADT) or in abstract classes. We use the algebraic specification to describe the ADT. Entities of the domain level are represented by the *concrete classes*. The latter are described from the abstract classes by derivation process. This process consists mainly in interpreting abstract data types by domains of values and each abstract class by set of the concrete classes (carriers). We illustrate the proposed formalism by examples from the urban domain.

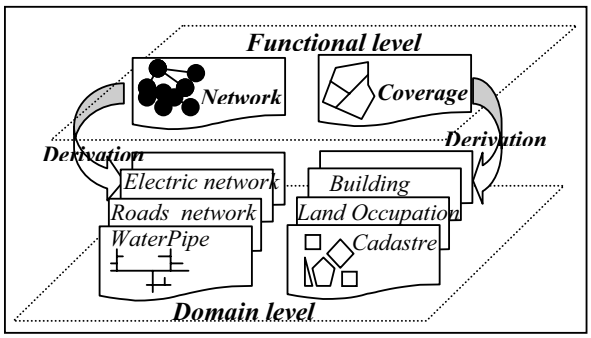

Figure. 1. Multi-leveled ontologies for urban applications

The remainder of the paper is organized as follows. Section 2 discusses background and related issues of ontologies and ontology mapping. To clarify the proposed formalism, sections 3 introduces some definitions and notations concerning the algebraic specifications. The proposed formalism is developed in sections 4 and 5 and it is illustrated by examples from the urban domain. Section 6 study the relations between ontologies. Section 7 presents some related work. Finally, section 8 concludes the paper.

2. Multi-layered ontologies and ontology mapping

In [6], the authors present a methodology to define multi-leveled ontologies for urban applications. This methodology is based on notions of abstraction and organization of the information in thematic layers within the spatial systems. Each thematic layer is characterized by a generic set of functionalities. For instance, the entities of the thematic layers (*roads networks*, *electric networks*, *water pipe networks*, etc.) share common functional features. So they can be described by a generic functional entity (called *network*) based on graph terminology and graph traversal operations which includes: inter-nodes distance evaluation, path cost optimization, path traversal, etc. This description is made without any implication of application domain. For example, the nature of objects that traverse the graph is less important at this level of abstraction. This information will be clarified at the lower level where these objects will be cars in the case of the road networks or the flows in the case of the drinking water system. Similarly, a two-dimensional spatial objet (called *coverage*) can be used to define the generic functional features of the entities of the thematic layers (*Cadastre, land occupation, buildings, parks,* etc.) including: surface computation, distance evaluation, adjacency test, etc. To meet the above design requirements, each layer is organized into two levels (see Figure 1) :

- **Functional level**: it contain generic entities including: *Node, Link, Surface*, etc. ; each entity is mainly represented by an abstract functional description in the generic ontologies.
- **Domain level**: it contains specific entities to a given domain (roads networks, Cadastre, land occupation, electric networks, etc.). Their descriptions are obtained from the generic entities by a derivation process. Those descriptions constitute the domain ontologies.

[9, 10] adopt an algebraic approach to descried ontologies and their relation. In this case, the ontologies are presented as logical theories: an ontology is defined as a pair $O=(S,A)$ where S is the ontological signature describing the vocabulary and A is a set of ontological axioms specifying the intended interpretation of the vocabulary in some domain of interest. The relations between ontologies are described by means of ontology morphisms. For more detailed descriptions of the field of ontology mapping and Semantic Integration Technologies, see the published survey [11,12].

3. Algebraic specifications : Background

In this section, we introduce some definitions and notations in order to clarify the proposed formalism. We use the algebraic specifications to describe the data types in the generic ontologies in the functional level. In general, algebraic specifications are used to capture the behavior of objects in a formal manner. It is possible to create complex types by using specifications of other. An important purpose of specifications is to organize types, values and operations [13, 14]. They are based on solid mathematical foundations and many methods can be applied to them. In the software engineering, they are used to describe the abstract data types and to prove that implementation is correct. They are also used in the other domain, in particularly in the spatial applications [15].

Definition 1 (Signature) : A signature is a pair (S, Σ) where :

- $S = \{ s_1, s_2, ..., s_n \}$ is a set of sorts. Each $s \in S$ denotes a domain of values.
- Σ a set of operations applicable to domains denoted by the sorts of S.

In fact, Σ is an indexed set (of operations) $\Sigma_{w,s}$:

$$\Sigma = \{ \Sigma_{w,s}, s \in S \}.$$

Each $\Sigma_{w,s}$ contain operations $\sigma_{w,S}$:

$\sigma_{w,S} : w \rightarrow s, s \in S$ *et* $w \in S^*$ with $w = (s_1, s_2, .. , s_n)$.

$\Sigma_{[],S}$ contains operations without arguments and returns a constant value.

When the sorts of S are organized by partial order relation, we say that (S, Σ) is order-sorted signature. Two types of relations can be distinguished: *Isa* relation and *Part-of* relation. The expression $(s_2 \leq s_1)$ indicates that the sort s_2 inherit all the specification and operations of the sort s_1 . The *Part-of* traduces the situation when one sort uses an another one.

One can define an algebra, called Σ-Algebra, relative to a signature (S,Σ). This consists in interpreting each sort of S by a domain of values and each operation by a function. Formally :

Definition 2 (Σ-Algebra):

Given a signature (S, Σ), a Σ-Algebra is an indexed set A, $A = \{ A_s , s \in S \}$ with an application I^A that associate each operation $\sigma_{w,s} \in \Sigma_{w,s}$ with a function $f_{w,s} = I^A(\sigma_{w,s})$ interpreting it:

$f_{w,s} : (A_{s1} x A_{s2} x .. x A_{sn}) \rightarrow A_s$ where $w = (s_1, s_2, .. , s_n)$

A_{si} is interpreting domain of the sort s_i for $i=1..n$.

Note that, $\forall s_i \in S$, $\forall s_j \in S$: $s_j \leq s_i \Rightarrow A_{si} \subseteq A_{sj}$ where A_{si} et A_{sj} are respectively interpreting domains of sorts s_i and s_j.

Definition 3 (Σ-Homomorphisme) : Given a signature (S , Σ) and two Σ-Algèbre A et B. Σ–Homomorphisme h from A to B is an indexed set of functions $\{ h_s : A_s \rightarrow B_s, s \in S \}$ such that, $\forall \sigma_{w,S} \in \Sigma_{w,S}$, $a_i \in As_i$ for $i= 1 .. n$:

$$h_s (f_{w,S} (a_1, a_2, ..., a_n)) = g_{w,S} (h_{s1} (a_1) , ... , h_{sn} (a_n))$$

where $f_{w,S} = I^A(\sigma_{w,S})$ and $g_{w,S} = I^B(\sigma_{w,S})$

Definition 4 (Algebraic specification) : An algebraic specification is a triple (S, Σ, E) where (S, Σ) is a signature, and E is a set of equations defining the behavior of the operations of Σ.

Definition 5 (Model) : A Model, called M_Σ , of an algebraic specification (S, Σ, E) is Σ-Algebra satisfying all equations in E.

The specification (S, Σ, E) describes the syntax of the abstract data types and the model M_Σ their semantics.

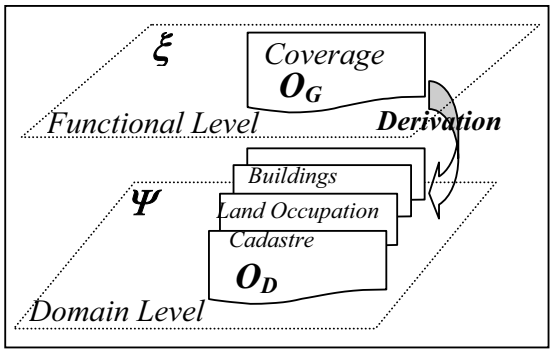

Figure. 2: Functional and domain levels

In the next section, we present a formalism to describe the multi-leveled ontologies for urban applications. These ontologies are organized in many layers. Each layer contains the entities that share common functional features and are described according to two levels of abstraction. In the functional level, generic ontologies O_G describe entities using abstract classes ξ. In the domain level domain ontologies O_D use concrete classes Ψ. Our ontologies specify entities with attributes to describe their proprieties and function describing their functional aspect. Axioms are used to define the semantics of attributes and functions.

4. Generic ontologies

4.1. Introduction

Formally, O_G is described by a set of abstract classes ξ described with their intensions. The classes of ξ are organized in hierarchy by the *Isa* ($\leq$) and *Part-of* relations. Each generic entity $e_G \in O_G$ is defined by abstract class $\alpha \in \xi$. The proprieties e_G are described by the attributes of α and the functions of α define the behavior of e_G. Constraints expressed by axioms can be used to describe semantics of attributes and functions of the abstract class α.

4.2. Formalization

Let's designate by T a set of abstract data types (ADT) defined by an algebraic specification (S, Σ, E) where S is a many order-sorted.
Let's designate by P_α a set of attributes of $\alpha \in \xi$ and by P the set of all attributes: $P = \cup_{\alpha \in \xi} P_\alpha$.
Note that, $\forall \alpha \in \xi,\ \forall \beta \in \xi, , \alpha \leq \beta \Rightarrow (P_\beta \subseteq P_\alpha)$

Let's designate by $Type_p$ an application that associate the attributes of classes of ξ with their types:

$$Type_p : \xi \times P \rightarrow \xi \cup T$$

$\forall \alpha \in \xi$, $\forall p \in P_\alpha$, two cases are possible :
- $Type_p (\alpha , p) = s \in T$: the attribute p take values in ADT specified by a sort s. Note that $Type_p (\alpha , p)$ can be a basic type (integer, real, string, etc.)
- $Type_p (\alpha , p) = \lambda \in \xi$, here the attribute p describe a relation (*Part-of*) between α and λ.

Let's designate by F_α a set of function of $\alpha \in \xi$ and by F the set of all functions: $F = \cup_{\alpha \in \xi} F_\alpha$. Note that, $\forall \alpha \in \xi, \forall \beta \in \xi, , \alpha \leq \beta \Rightarrow (F_\beta \subseteq F_\alpha)$

Let's designate by Π_α a set of arguments of functions of F_α for $\alpha \in \xi$, and by Π the set of all arguments: $\Pi = \cup_{\alpha \in \xi} \Pi_\alpha$
Each function $f \in F_\alpha$ is defined by its profile as follow: $f (\{ a_k : t_K \}) \rightarrow a_r : t_r$
with $a_k \in \Pi_\alpha$, $t_K \in (\xi \cup S)$, for $k = 1 \ldots m$, $a_r \in \Pi_\alpha$, $t_r \in (\xi \cup S)$

Let's designate by $Type_a$ an application that associate the argument $a \in \Pi_\alpha$ of $f \in F_\alpha$ with their types: $Type_a : \xi \times F \times \Pi \rightarrow \xi \cup T$
Two cases are possible : $Type_a (\alpha , f, a) = s \in T$ or $Type_a (\alpha , f, a) = \lambda \in \xi$.

Let's designate by *CardMin* and *CardMax* applications that define the minimum and maximum cardinality constraints of the attribute p of $\alpha \in \xi$:
$CardMin : \xi \times P \rightarrow N$; $CardMax : \xi \times P \rightarrow N$, N is a set of natural numbers (integer). Note that :

If $CardMin (\alpha , p) = 1$ and $CardMax (\alpha , p) = 1$; p is a mono value attribute
If $CardMin (\alpha , p) = 0 / 1$ et $CardMax (\alpha , p) = n \in N$; p is a multi value attribute
$CardMin (\alpha , p) = n1 \in N$ et $CardMax (\alpha , p) = n2 \in N$; correspond to general case.

Formally, generic ontology is described, to a high level of abstraction, by a hierarchy of abstract classes ξ. Each class $\alpha \in \xi$ is described its intension. This latter contains a set P_α of attributes and a set $F\alpha$ of functions. Each function $f \in F_\alpha$ is represented by its profile defining the types of its arguments. Those latter and attributes types are defined by ADTs or by abstract classes.

Definition 6 (Intension $(I_\xi)_\alpha$) : The intension of abstract class $\alpha \in \xi$, noted $(I_\xi)_\alpha$, is given by :

$$(I_\xi)_\alpha = < P_\alpha ; Type_{p,\alpha} ; CardMin_\alpha ; CardMax_\alpha ; F_\alpha ; Type_{a,\alpha} >$$ such as,

$$\forall p \in P_\alpha,\ \forall f \in F_\alpha,\ \forall a \in \Pi_\alpha :$$
$$Type_{p,\alpha} (p) = Type_p (\alpha , p) ;$$
$$Type_{a,\alpha} (f, a) = Type_a (\alpha , f, a),$$
$$CardMin_\alpha (p) = CardMin (\alpha , p) ;$$
$$CardMax_\alpha (p) = CardMax (\alpha , p)$$

Note that : $\forall \alpha \in \xi,\ \forall \beta \in \xi,: \alpha \leq \beta \Rightarrow (I_\xi)_\beta \subseteq (I_\xi)_\alpha$

Definition 7 (Generic ontology): Generic ontology O_G is defined by :

$$O_G = < T ; \xi ; I_\xi ; \Lambda_{\xi F} > \text{ where}$$

- T : Set of ADTs specified by algebraic specification (S, Σ, E),
- ξ : Hierarchy of abstract classes described in intension by I_ξ
- I_ξ : Set of the intensions of abstract classes, $I_\xi = \cup_{\alpha \in \xi} (I_\xi)_\alpha$ where $(I_\xi)_\alpha$ is the intension of $\alpha \in \xi$
- $\Lambda_{\xi F}$: An indexed set , $\Lambda_{\xi F} = \{ (\Lambda_{\xi F})_\alpha$ pour $\alpha \in \xi \}$ where $(\Lambda_{\xi F})_\alpha$ is the set of axioms relative to the functions F_α of $\alpha \in \xi$.

4.3. Example

For example, consider the set of generic spatial entities : { *Position, Node, Link, Oriented-Link* } , described as follow:

Node
position : Position
isolated : Boolean
Get-position(N : Node) → P : Position
Is-Isolated(N : Node) → B : Boolean
... etc.

Position

x : Number
y : Number
Get-x (P : Position)→X : Number
Get-y (P : Position)→Y : Number
Put-x (X : Number)→P : Position
Put-y (X : Number)→P : Position
Identique (P1: Position, P2: Position)→I : Boolean
Identique (p1, p2) ⇒
(Get-x (p1) = Get-x (p2)) & (Get-y (p1) = (Get-y (p2))
... etc.

Link

n1 : Node
n2 : Node
Oriented : Boolean
Weight : Number
Creat-Link(N1 :Node, N2 :Node)→A :Link
Get-Weight(A : Link)→P :Number
... etc.

Oriented-Link ≤ Link

orientation : Orientation
Get-orientation(A :Oriented-Link)→O :Orientation
... etc.

According to the definition 7 above, we have:

ξ = { *Position, Node, Link, Oriented-Link, etc.* } ,
T = { Number, Boolean, etc. }
P_{Link} = { *n1, n2, Oriented, Weight* } ,
$Type_p$ *(Link, n1) = Node*
$Type_p$ *(Link, Weight) = Number* ,
$F_{Position}$ = { *Get-x , Get-y, Put-x, Put-y, Identique* },
$Type_a$ *(Position ,Get-x, X) = Number*
$Type_a$ *(Position , Put-x, P) = Position.*

5. Domain ontologies

5.1. Introduction

The domain level contains entities that are specific to a given domain. Those entities are organized in the thematic layer. For instance, the entities of water networks, roads networks, electric networks, etc. are regrouped in the same layer. For a given layer, we use domain ontology O_D to describe the entities of each thematic. Formally, the domain ontologies O_D are represented by a set of concrete classes Ψ. Those classes are described by their intensions and interpreted by their extensions. The classes of Ψ are organized en hierarchy.

5.2. Formalization

Let's designate by Ψ a set of the concrete classes. Note by Ψ_ξ the set of classes of Ψ that are directly derived from classes of ξ ($\Psi_\xi \subset \Psi$).
Let's designate by Ω_c the set of instance of $c \in \Psi$ and by Ω the set of all instances, $\Omega = \cup_{c\in\Psi}\Omega_c$.
Let's designate by Φ an application that associate each class $c \in \Psi$ with its instances: $\Phi(c) = \Omega_c$:

$$\Phi : \Psi \rightarrow \Omega, \text{ such that}$$
$$\forall b \in \Psi \quad \forall c \in \Psi : b \leq c \Rightarrow \Phi(b) \subseteq \Phi(c)$$

Let's designate by Γ_ξ the application defined from ξ to Ψ. This application associates an abstract class $\alpha \in \xi$. with a concrete class $a \in \Psi$.
The class $a, \Gamma_\xi(\alpha) = a,$ represents the interpretation of de α or its derivative.
Let's designate by $\alpha^\Gamma \subset \Psi_\xi$ the set of a concrete class that directly derived from $\alpha \in \xi$:

$$\alpha^\Gamma = \cup\Gamma_\xi(\alpha) \text{ with } \alpha^\Gamma \subset \Psi_\xi$$

For instance, if α = *Link* then α^Γ = { *Road-Link, Electric-Link, etc.* } where *Road-Link* and *Electric-Link* are classes of Ψ. Note that, $\forall c \in \alpha^\Gamma$, $P_c = P_\alpha$ *et* $F_c = F_\alpha$. That is the class $c = \Gamma_\xi(\alpha)$ inherit the attributes and functions of $\alpha \in \xi$. In this case we have : $P = \cup_{c\in\Psi_\xi} P_c$; $F = \cup_{c\in\Psi_\xi} F_c$. Similarly, we have, $\forall c \in \alpha^\Gamma$, $\forall p \in P_c$: *CardMin(c, p) = CardMin(*α*, p) ; CardMax(c, p) = CardMax(*α*, p)*

Let's designate by Γ_T the application that associate each sort $s \in S$ with its interpreting domain D_s :

$$\Gamma_T : S \rightarrow M_\Sigma$$

M_Σ is (definition 5) the model of algebraic specification (S, Σ, E) that is used to describe T in the functional level.

Let's designate by *Domp* an application that associate an attribute $p \in P_c$ for $c \in \Psi$ with its values domain noted *Domp (c,p)* :
$\Psi x P \rightarrow D$, where $D = \Omega \cup M_\Sigma$ such that
$\forall c \in \alpha^\Gamma$, $\forall p \in P_c$:
Domp (c, p) = $\Phi(\Gamma_\xi(\beta))$ if $type_p(\alpha,p) = \beta \in \xi$.,
Domp (c, p) = D_s if $type_p(\alpha,p) = s \in S$, with $D_s = \Gamma_T(s)$.

Note that, $\forall c \in \Psi$, $\forall d \in \Psi$:
$c \leq d$ & $p \in P_d \Rightarrow$ *Domp(c, p)* $\subseteq$ *Domp(d, p)*
$o \in \Phi(c)$ & $p \in P_c \Rightarrow$ *p(o)* $\in$ *Domp(c, p)* ;
p(o) is the value of the attribute $p \in P_c$ for the instance $o \in \Phi(c)$.

Let's designate by Π_c a set of arguments of F_c for $c \in \Psi$ and by Π the set of all arguments: $\Pi = \cup_{c\in\Psi}\Pi_c$.

Let's designate by *Doma* an application that associate an argument $a \in \Pi_c$ of $f \in F_c$ with its values domain, noted *Doma (c,p)* :

Doma : $\Psi \times F \times \Pi \rightarrow D$ such that $\forall c \in \alpha^{\Gamma}$, $\forall f \in F_c$, $\forall a \in \Pi_c$:

$Doma(c, f, a) = \Phi(\Gamma_{\xi}(\beta))$ if $type_a(\alpha, f, a) = \beta \in \xi$.,

$Doma(c, f, a) = D_s$ if $type_a(\alpha, f, a) = s \in S$,

with $D_s = \Gamma_T(s)$.

Definition 8 (Intension $I_{\Psi,c}$) : The intension of a concrete class $c \in \Psi$, noted $I_{\Psi,c}$, is given by :

$$I_{\Psi,c} = < P_c ; Domp_c ; CardMin_c ; CardMax_c ; F_c ; Doma_c >$$

where

$\forall p \in P_c$, $\forall f \in F_c$, $\forall a \in \Pi_c$:

$Domp_c(p) = Domp(c, p)$;

$Doma_c(f, a) = Doma(c, f, a)$;

$CardMin_c(p) = CardMin(c, p)$;

$CardMax_c(p) = CardMax(c, p$

According to the above definition, we have, $\forall c \in \Psi$, $\forall d \, c \in \Psi$:

$$c \leq d \Rightarrow (I_{\Psi,d} \subseteq I_{\Psi,c})$$

Definition 9 (Domain ontology): Domain ontology is defined by :

$$O_D = < M_{\Sigma} ; \Psi ; I_{\Psi} ; \Gamma ; \Lambda_{\Psi,F} ; \Lambda_{\Psi,P} >$$ where

M_{Σ} : Model of algebraic specification *(S,Σ, E)*,

$M_{\Sigma} = \cup_{s \in S} D_s$

Ψ: Hierarchies of concrete classes described by its intension I_{Ψ}

I_{Ψ}: Set of the intensions of classes of Ψ, $I_{\Psi} = \cup_{c \in \Psi} I_{\Psi,c}$

Γ: Derivation function $\Gamma = (\Gamma_T, \Gamma_{\xi})$

$\Lambda_{\Psi,F}$: An indexed set : $\Lambda_{\Psi,F} = \{ (\Lambda_{\Psi,F})_c$ pour $c \in \Psi \}$ where $(\Lambda_{\Psi,F})_c$ is a set of axioms associated to functions F_c of $c \in \Psi$.

$\Lambda_{\Psi,P}$: An indexed set : $\Lambda_{\Psi,P} = \{ (\Lambda_{\Psi,P})_c$ pour $c \in \Psi \}$ where $(\Lambda_{\Psi,P})_c$ is a set of axioms associated to attributes P_c of $c \in \Psi$.

The axioms of $\Lambda_{\Psi,P}$ can be expressed in different manners. One can use rules. For instance, the following rule can be used to define dependency between two attributes $p \in P_c$ et $p' \in P_c$ of the same class $c \in \Psi$: $\forall o \in \Phi(c)$, $p(o) = v \Rightarrow p'(o) = v'$.

One can also use an equation to define constraints on values of attributes. For instance, the equation below constraints values of the attribute $Age \in PYoung$ such *that (Young ≤ Individu):*

$\forall o \in \Phi(Young)$, $Age(o) <= 35$.

5.3. Example

Figure 3, illustrate an example of domain ontology. Here we have:

$Node\Gamma$ = { *Road-Node, Water-Node,, etc.* },

$Public \leq Place \leq Road\text{-}Node$;

$Manual \leq Vane \leq water\_Node$

Note that classes of Ψ are hierarchically organized. The top of this hierarchy are classes of $\alpha\Gamma = \cup \Gamma\xi(\alpha)$, $\alpha \in \xi$..

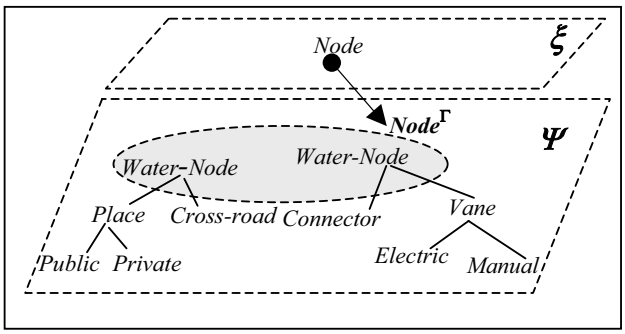

Figure 3. Example of domain ontology

6. Inter-ontology relationships

Ontology mapping is important when deal with multiple ontologies.

Many Inter-ontology relationships can be expressed [28, 20]. These relations can be defined at both functional and domain level. Figure 4 illustrates possible examples of relationships between entities.

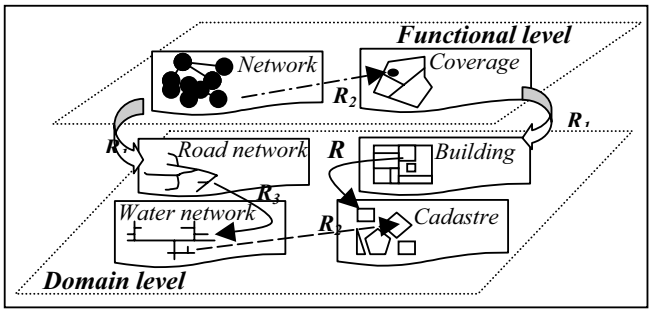

Figure 4. Examples of Inter-ontology relationships

Relation (R1) : correspond to the derivation process. This relation enables us to maintain link between generic entities pertaining to the functional level and the specific entities of domain level that derives from them.

Relation (R2) : correspond mainly to topological link amongst spatial entities (inclusion, intersection, etc.). For example, this type of relationships permit to maintain a link between entities pertaining to a water network

(running water pipe) and a entity of Cadastre planing (zone traversed by this pipe).
Relation (R_3) : this type of relation correspond to the following different situations:
Two entities of differences theme can cover the some spatial area. It may be the case, for instance, between road entity (an avenue) and an electric network entity (electric wire buried along this avenue) or the case of a building situation (the building itself) and the area on which it is build.

To formalize those relations an algebraic approach can be possible as described in [28]. In order to just point the problem, let's consider a simple case. For $\alpha \in \xi$, $p \in P_\alpha$ such $type_p(\alpha,p) = s \in S$, one can define a Σ-homomorphisme *h* as follow:

$$h : D_s \rightarrow D'_s \text{ with}$$
$$Domp(c,p)=D_s ;$$
$$Domp(d,p)=D'_s ; c \in \alpha^\Gamma \text{ and } d \in \alpha^\Gamma$$

To illustrate this by an example, consider the abstract class α = *Person*, and the attribute $p \in P_\alpha$, p = *sexe*. Given α^Γ = { *Individu ,Human* } and suppose that:

$$Domp\ (\ Individu,\ sexe\ ) = D_{Sexe}$$
$$= \{\ 'Masculine',\ 'Feminine'\ \}$$
$$Domp\ (\ Humain,\ sexe\ ) = D'_{Sexe}$$
$$= \{\ 'Man',\ 'Woman'\ \}$$

A possible Σ-homomorphisme h_{sexe} can be defined as follow:

$$h_{sexe}\ ('Masculine') = 'Man' \text{ and}$$
$$h_{sexe}\ ('Feminine') = 'Woman'$$

7. Related word

The work presented in this article concerns the problem of description of ontologies and their relations. It is partially inspired by the work presented in [6,9]. Like [6] we consider the case of an urban application and multi-layered spatial ontologies but in this paper we presents a formalism based on algebraic approach to describe and study the such ontologies. Unlike [9] that descried, in general an theoretical manner, an algebraic approach to describe ontologies and their relation, we use the such approach to deal with a concrete example from urban application.

8. Conclusion and future work

In this paper, we present a formalism to describe the multi-leveled ontologies for urban applications. The proposed formalism is based on the fact that the spatial systems can be viewed as comprising several abstract layers, each defining a generic set of functionalities. Each layer contains the entities that share common functional features and are described according to two levels of abstraction. In the functional level, generic ontologies describe entities using abstract classes. In the domain level ontologies are described by concrete classes. Our ontologies specify entities with attributes to describe their proprieties and function describing their behavior. Axioms are used to precise the semantics of attributes and functions. We use the ADT to describe the abstract classes. Process derivation allows us to describe the domain specific entities from the generic entities. It consists mainly in interpreting in the domain level the ADT by their models and each abstract class by a set of concrete classes. The latter are the top of the hierarchy of the classes in the domain level.
Σ–homomorphisme can be used to make relations between entities from the same level. Other types of relations, that are not developed in this paper can be used. Its an interesting issue to explore and to investigate. Another issue to explore is the study of the implementation of proposed formalism using an XML language like OWL. This will enable us to study the limits of this formalism. It is also interest to seek who this framework to describe the multi-leveled ontologies can be use in the other applications, in particularly, the e-learning domain.

Bibliographie

1. Semantic Integration: A Survey Of Ontology-Based Approaches by N.F.Noy. SIGMOD Record, Special Issue on Semantic Integration, 33 (4), December, 2004
2. Kalinichenko L.A., Missikoff M., Schiappelli F., Skvortsov N. Ontological Modeling Proceedings of the 5th Russian Conference on Digital Libraries RCDL2003, St.-Petersburg, Russia, 2003
3. Mizoguchi, R., Tutorial on ontological engineering - Part 2: Ontology development, tools and languages. New Generation Computing, OhmSha & Springer, Vol.22, No.1, pp.61-96, 2004
4. Distributed Reasoning Services for Multiple ontologies. Technical Report DIT-04-029 Department of Information and Communication Technology, 2004
5. N.Guarino: Understanding, building and using ontologies. International Journal of Human and Computer Studies, 46(2/3), 293 – 310
6. D. Beslimane, E. Leclercq, M. Savonnet, M.N. Terrasse, K. Yétongnon: On the definition of generic multi-layered ontologies for urbain applications. Computers,Environment and Urban Systems, 24 (2000) 191-124
7. T. Gruber: A translation approach to portable ontology specifications. International Journal of Knowledge acquisition for knowledge-based systems, 2(5), 199-220, 1993.

8. Y.Kalfoglou, Exploring ontologies, ,The Handbook of Software Engineering and Knowledge Engineering - 2001
9. Trevor J. M. Bench-Capon, Grant Malcolm: Formalising Ontologies and Their Relations. In Bench-Capon T. et Soda
10. Y.Kalfoglou, M.Schorlemmer "Information Flow based ontology mapping", In Proceedings of the 1st International Conference on Ontologies, Databases and Application of Semantics (ODBASE'02), Irvine, CA, USA, October 2002
11. Y.Kalfoglou, M.Schorlemmer , Ontology mapping: the state of the art, ,The Knowledge Engineering Review – 2003
12. Y. Kalfoglou, B. Hu, D. Reynolds and N. Shadbolt. Semantic Integration Technologies Survey. CROSI project, 6th month deliverable. University of Southampton, Technical Report, E-Print No #10842. May, 2005
13. Simone Veglioni: Classifications in algebraic specifcations of abstract data types, Technical Report TR-7-96, Programming Research Group, University of Oxford, Agust 1997, 28 pp..
14. Goguen J. A., R. Diaconescu. An Oxford Survey of Order Sorted Algebra. Mathematical Structure in Computer Science, Vol. 4, pp. 363-392, Septembre 1994.

15. Frank, A.U., & Kuhn, W. 1995: Specifying Open GIS with Functional Languages. In Advances in Spatial Databases (4th International Symposium, SSD'95 in Portland, ME). (Egenhofer, M.J., & Herring, J.R., eds.), Lecture Notes in Computer Science, Vol. Lecture Notes in Computer Science Vol. 951, Springer-Verlag, pp: 184-195.