S. Khouri, L. El Saraj, L. Bellatreche, B. Espinasse, N. Berkanil, S. Rodier, T. Libourel,
« CiDHouse: Contextual SemantIc Data WareHouses », DEXA 2013, Prague, Czech Republic, August 26-29, 2013.

CiDHouse: Contextual SemantIc Data WareHouses

Selma Khouri[1],[3], Lama El Saraj[2],[4], Ladjel Bellatreche[3], Bernard Espinasse[2], Nabila Berkani[1], Sophie Rodier[4]

[1] ESI, Algiers, Algeria
`(s_khouri, n_berkani)@esi.dz`
[2] LSIS, Marseille, France
`(firstname.lastname)@lsis.org`
[3] LIAS/ISAE-ENSMA, Futuroscope, France
`(selma.khouri, bellatreche)@ensma.fr`
[4] Assistance publique des Hpitaux de Marseille, France
`(lama.elsaraj,sophie.rodier)@ap-hm.fr`

Abstract. Dealing with contextualized data is a key challenge in different fields of information systems, databases and data warehouses ($\mathcal{DW}$). Nowadays, $\mathcal{DW}$ Systems are often mono-context. However, in real life applications, $\mathcal{DW}$ indicators are shared by many users from different profiles. For example, in the medical domain, users can be doctors, researches, nurses, computer scientists, etc. They need comprehensive results adequate to their context. To tackle this challenge we propose to use ontologies to treat contextualization problem at the semantic level. Thereby, we offer an approach that makes DW systems multi-contextual and able to personalize results based on user's context. In this paper, we present a novel approach based on ontologies for designing multi-contextual $\mathcal{DW}$. We propose an ontology formalism that incorporates the contextualization concepts. We specify our approach that takes contextualization into account from the conceptual level. We validate our proposal using a real case study from the medical domain.

1 Introduction

Nowadays, the data warehouse technology becomes an incontestable technology and tool for businesses and organizations to make strategic decisions. Data warehouse ($\mathcal{DW}$) is considered as a pillar of the integration industry widely developed in last two decades and recently in the big data field [7]. A $\mathcal{DW}$ is defined as a stepwise information flow from information sources owned by heterogeneous services and departments through materialized views towards analyst clients. One of the difficulties of building a DW application is handling the heterogeneity of information sources. Ontologies play an important role to reduce this heterogeneity and to reason on the ontological concepts. Note that domain ontologies have been widely developed in several domains such as medicine, environment, engineering, etc. This development motivates the $\mathcal{DW}$ community to consider it in the design steps of the warehouse applications, especially in the conceptual modeling and ETL phases [22, 4]. During the conceptual phase the ontologies may represent the global schema of the $\mathcal{DW}$ [3, 6]. Some other works proposed then to attach an ontology (usually called local ontology) to each source participating in the $\mathcal{DW}$ contruction, and to define mappings between the global and local ontologies [14, 25]. The $\mathcal{DW}$ considering sources embedding ontologies in their repository (usually called semantic databases (SDB)) correspond to this architecture. Two integration scenarios can be defined: (i) correspondences between global and local ontologies are defined *a priori* at the design time of SDB. In such case, the integration process is simply assimilated to an integration of mappings. Designers agree to make efforts when designing the sources in order to get a '*free*' ETL process. (ii) Correspondences are discovered *a posteriori* either *manually* or *automatically*, which is an issue related to the domain of schema and ontology *matching/alignment*. Once the mappings discovered, the integration process resembles to the first scenario.

In the first era of the $\mathcal{DW}$, the business applications were mono-contextual. A $\mathcal{DW}$ is exploited by multi-context users, for example in the healthcare domain users are doctors, researchers, directors, etc. Consequently, users' needs and interpretation depend on the context. For instance, the computation of the stay duration of a patient depends on users' context of analysis. Consequently, users' needs and interpretation depend on the context. A context is a general term used in several domains [19]. The importance of a context representation for the semantic integration of heterogeneous databases was already underlined by a number of researchers in multi-database systems, both at the schema definition level [12] and at the property value level [11]. To ensure the feasibility to reach a consensus on an ontology definition, ontologies such as PLIB minimize context sensitivity [18]: (1) explanation of the definition context is effectuated by associating to each property the higher significant class and to each class the applicable properties to each class; (2) explanation of values context is done by associating to each context-dependent property value its context evaluation represented as pairs of context parameter-value, (3) explanation for scaling of values is done at the schema level, by associating to each quantitative property type both a unit and a dimensional equation, and (4) to avoid context influences in the choice of properties associated to a class, each class is associated, at the ontology level, to essential properties for its instances.

We differentiate between two general context types: (i) internal and (ii) external context [20]. Internal context captures conditions that involve the data items stored in the database for which preferences are expressed. External context involves conditions outside the database. Common types of external context include the computing context (e.g., network connectivity, nearby resources), the user context (e.g., profile, location), the physical context (e.g., noise levels, temperature) and the time context [20]. Actually, to extract contexts data mining, techniques have been used [24]. These studies assume that the $\mathcal{DW}$ is build.

There is a paradox in the companies, they need to construct a $\mathcal{DW}$ to store data from heterogeneous sources but the context heterogeneity of end users is not taken into consideration during the construction time. The $\mathcal{DW}$ end-users have a rich knowledge warehouse regarding the context, measure unity, etc. that should be formalized and integrated in the ontologies. A couple of research efforts have been proposed contextual ontologies. We can cite for instance CONON: contextual ontologies for pervasive computing environments [26].

In this work, we present a novel approach based on ontologies to propose a Contextual-SemantIc Data WareHouses (CiDHouse) design. We tested our approach in the healthcare domain in collaboration with the public hospitals of Marseille - France. To the best of our knowledge, we are the first to propose an ontology-based approach for multi-context $\mathcal{DW}$ covering the conceptual, logical and ETL phases.

The paper is organized as follows. Section 2 presents the related work of contextual $\mathcal{DW}$ design. Section 3 explains the intuition of our proposal. Section 4 describes the design method we propose. Section 5 presents a case study validating our proposal. Section 6 presents the case tool supporting our proposal. Section 7 concludes the paper by summarizing the main results and suggesting future work.

2 Related Work

Historically, the notion of context was used to indicate "the part of speech around the content that can explain its significance". Most of the times contents needs to be followed by knowledge about the context to help users in the interpretation and analysis. Otherwise, each user interprets the content in function of his experience, culture or domain, etc. The context has been studied in various fields of computer science such as artificial intelligence (AI), information retrieval (IR), databases, machine learning, knowledge representation, $\mathcal{DW}$, etc. According to the literature, we classify the state of the art studies in two categories: (i) naive approaches, (ii) algorithmic approaches. Naive approaches are based on logic properties annotated by the context. At the conceptual level of the $\mathcal{DW}$, Pitrach *et al.* [19] considers context for enhancing the flexibility and the expressivity of hierarchies to overcome generalization problem. This approach is limited on data hierarchies depending on context of analysis, e.g., the attribute TensionCategorie (High, Low or Normal) is context dependent. This attribute depends on contextualizing attributes such as AgeCategorie, Smoker, and TensionValue. At the exploitation level of the $\mathcal{DW}$ Perez *et al.* [16] proposes a documentation for OLAP queries that takes the context into consideration. Also, at the exploitation level Garrigos *et al.* [9] propose an approach to personalize OLAP schema for each decision maker taking into account the ever-changing user characteristics, context, requirements and behavior. Algorithmic approaches propose algorithms to adapt the results of a system to a context. Kostas *et al.* [23] propose a computation of contextual group recommendations based on a subset of preferences of users that present the most similar behavior to the group. The proposed approaches give solutions for the two phases of the $\mathcal{DW}$ the conceptual and the exploitation level. In order to compare the previous approaches, we identified the following criteria:

– C1 : Multi-context exploitation techniques. The exploitation of the data ware-house for multi-context uses provides to each user a response adapted to his needs. This criterion identifies if the approach proposes techniques to facilitate multi-context $\mathcal{DW}$ exploitation (it could be at the conceptual level or at the physical level).
– C2: Multi-context measures affected by external entities to the $\mathcal{DW}$. The measures could be affected by external criteria to the $\mathcal{DW}$, like domain of analysis. This criterion identifies if the approach takes in consideration the semantic external criteria to the $\mathcal{DW}$.
– C3: Multi-context measures influenced by entities internal to the $\mathcal{DW}$. Contextual hierarchies influencing measure analysis. This criterion identifies if the contextual hierarchies are considered.
– C4: Knowledge about the contents. Having knowledge about the contents help the users to interpret and analyze contents. This criterion identifies if the approach gives knowledge about contents.
– C5: Semantic variation. The interpretation, the identification, the expression of measures and dimensions could differ from a user to another, depending on users-context of analysis. This criterion identifies if the approach takes in consideration the semantic variation in a multi-context data warehouse domain.

The following table lists the previous approaches in order by approach category, the $\mathcal{DW}$ phases, approach reference, approach description, and criterion (C1,C2,C3,C4, C5).

$\mathcal{DW}$ phase	Reference	Description	C ategory	C1	C2	C3	C4	C5
Conceptual level	CiDHouse	Ontology-based for multi-context $\mathcal{DW}$	Ontological	x	x	x	x	x
Conceptual level	[19]	Flexibility and expressivity of hierarchies	Naive	x		x		
$\mathcal{DW}$ exploitation	[9]	Visualization of personalized multidimensional model	Naive	x	x			
$\mathcal{DW}$ exploitation	[16]	Contextualization with document	Naive	x		x	x	
$\mathcal{DW}$ exploitation	[23]	Algorithm to adapt system to the context	Algorithmic	x	x			

Table 1: Related work comparison

These approaches are situated at the conceptual and the exploitation level of the $\mathcal{DW}$. They propose solutions for multi-context uses of a $\mathcal{DW}$. They try to extract context using algorithms; or assign context to different entities of the data warehouse. Our approach is situated at the conceptual level. We propose a multi-context data warehouse based on user requirements, multi-context and data sources. The advantage of our approach compared with the listed approaches is that we consider the contextualization at the conceptual level. The additional characteristics of our approach to the existing approaches at the conceptual level: take in consideration multi-context measures affected by external entities to the $\mathcal{DW}$, gives knowledge about the contents, considers Semantic variation. We extended our ontology-based conceptual approach for $\mathcal{DW}$ [4] to help users to model a multi-context $\mathcal{DW}$ (multidimensional schema). This novel approach, facilitate the modeling and the exploitation of the $\mathcal{DW}$.

3 Intuition of our proposal

The classic $\mathcal{DW}$ model does not take contextual semantics in consideration. To resolute semantic problems ontologies have been used in several domains information systems, genetics, medicine, etc., e.g., NIFSTD a modular set of ontologies for the neuroscience domain, Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT), BMO for e-business model ontology, DO disease ontology that relates concepts such as the International Classification of Diseases (ICD) version 9 and 10 (used also for T2A diseases coding), MeSH, and UMLS. In general, ontologies are based on models that describe the absolute meaning of things in a domain and don't take in consideration particular uses of these things. To face this problem, several works have proposed ontology models to represent particular uses knowledge, .e.g., Pierra [17] *et al.* relates ontologies to the context, he proposed the PLIB ontology model.

In the $\mathcal{DW}$ domain ontologies have proved their utility to resolute semantic problems. Several works was proposed in different phases of the $\mathcal{DW}$, at the conceptual level [5], ETL [4], OLAP queries [15].

We present a novel approach based on ontologies to propose a Contextual-SemantIc Data WareHouses (CiDHouse) design. The advantage of our approach is that the context is taken in consideration during the conceptual, logical and ETL phases. Thus, at the exploitation level OLAP queries based on CiDHouse could be multi-contextual.

4 Case study

To illustrate the problematic related to our approach, we present a simplified medico-economic case study for healthcare establishment. The case study contains information about sojourn duration, coding exhaustiveness rate for each patient per Diagnosis Related Groups (used for the T2A system, to homogenize medico-economic groups), per time dimension (day, month, year), per institution structure (medical unit, service, pole, healthcare institution), per weight (important for the new born), per age and per international classification of diseases (used for the T2A system, for sojourn diseases coding). This domain is formally defined by an ontology that we constructed and validated with the hospital staff of the French healthcare institution of the public hospitals of Marseille. An extract from the ontology is illustrated in figure 2.

To give visibility about the context we'll present the diagnosis related groups (DRG). DRG is a classification system for hospital, invented at the United States of America (USA) by Pr Fetter, Yale University. DRG is the base for pricing activity (known as T2A in the French system) employed by the insurance to finance healthcare establishment. This financing system is applied since 80s in USA and recently by many others countries over the world. Our $\mathcal{DW}$ must respond to different users' needs to follow up and evaluate healthcare establishment activity. To clarify our approach we will take two examples:

The first one, an indicator inspired from the set of indicators provided by the national agency for performance support (ANAP), concerns "coding exhaustiveness rate" in the medico-economic context:

- Objective: Reduce the number of unbilled records and to improve the T2A income
- Challenge: Avoid billing blockage and maximize the income of the "T2A".
- Unit of measure %
- Methodological point: check completeness of data 15 days after the exit day and 30 days after the exit day
- Collector of data: Department of medical information (DIM)

Several data are coded in the health care establishment for example, medicaments, T2A data, etc. The results that respond to coding exhaustiveness rate should be adapted to the context of the user.

The second example concerns the sojourn duration. For example, "Sojourn duration average per DRG per service and per month". In real life, a service chief could find that measures values are incoherent or insufficient to the reality, because sojourn duration is calculated differently from a context to another. To clarify our example we'll present the calculation method (exceptions are not considered):

- We define DS = Δ (entry date, exit date)
- In the medico-economic context, sojourn duration = DS
- In the statistic context, sojourn duration = DS + 1

Unfortunately, responding to queries adapted to the context, of analysis, is not possible with the classic $\mathcal{DW}$ model. This problem could be resolved with the CiDHouse approach.

5 Our Proposal

Our $\mathcal{DW}$ design follows the hybrid approach, where data sources and user requirements are considered as inputs of the method. Another characteristic of our proposal is that it exploits the presence of ontologies. Our proposal is based on four foundations that we discuss in the next sections:

- F_1: Formalization of ontologies integrating context;
- F_2: User requirements defined on multi-contextual ontologies;
- F_3: Definition of the integration framework for integrating sources containing contextual data;
- F_4: Definition of the Contextual semantic Data wareHouse (CiDHouse) by following a design method including five design steps: requirements definition, conceptual design, logical design, ETL and physical design.

5.1 F_1: Formalisation of ontologies integrating context

Different languages have been defined to describe ontologies. OWL language is the language recommended by W3C consortium for defining ontologies. Description Logics (DLs) [2] present the formalism underlying OWL language. We thus use DLs as a basic formalism for specifying the framework.

In DL, structured knowledge is described using *concepts* denoting unary predicates and *roles* denoting binary predicates. Concepts denote sets of individuals, and roles denote binary relationships between individuals. Two types of concepts and roles are used: *atomic* and *concept descriptions.* Concept descriptions are defined based on other concepts by applying suitable *DL* constructors (eg. intersection, value restriction, limited existential quantification, etc), equipped with a precise set-theoretic semantics.

A knowledge base in DL is composed of two components: the $TBox$ (Terminological Box), and the $ABox$ (Assertion Box). The $TBox$ states the *intentional* knowledge of the modeled domain. Usually, terminological axioms have the form of inclusions: C $\sqsubseteq$ D (R $\sqsubseteq$ S) or equalities: C $\equiv$ D (R $\equiv$ S) (C,D denote concepts, R,S denote roles).

For example, in the ontology model of Fig. 2 representing the hospital ontology, the concept *Hospital* can be defined as an *Organization* by specifying the axiom: *Hospital* $\sqsubseteq$ *Organization*. The ABox states the *extensional* knowledge of the domain and defines assertions about individuals. Two types of assertions are possible: concept assertions (Eg. *Patient*(Patient#1)) and role assertions (e.g. *Affected*(Patient#1, Service#1)).

Based on these definitions, an ontology O is formally defined as follows:
$O :< C, R, Ref(C), formalism >$, such that:

- C: denotes *Concepts* of the model (atomic concepts and concept descriptions).
- R: denotes *Roles* of the model. Roles can be relationships relating concepts to other concepts, or relationships relating concepts to data-values.
- Ref : $C \rightarrow$ (*Operator*, *Exp*(C,R)). Ref is a *function* defining terminological axioms of a DL TBox. *Operators* can be inclusion ($\sqsubseteq$) or equality ($\equiv$). *Exp(C,R)* is an expression over concepts and roles of O using constructors of DLs such as union, intersection, restriction, etc.
- $Formalism$: is the *formalism* followed by the global ontology model like RDF, OWL, etc.

We extend this formalization for handling contextual information. Contextual information is described in literature as dependence relationships between contextualizing properties and contextualized properties. For example, the *Hospitalization duration* property is dependent on the *Medical Unit* and *IdPatient* properties. *Medical Unit* and *IdPatient* are contextualizing properties and *Hospitalization duration* is a contextualized property.

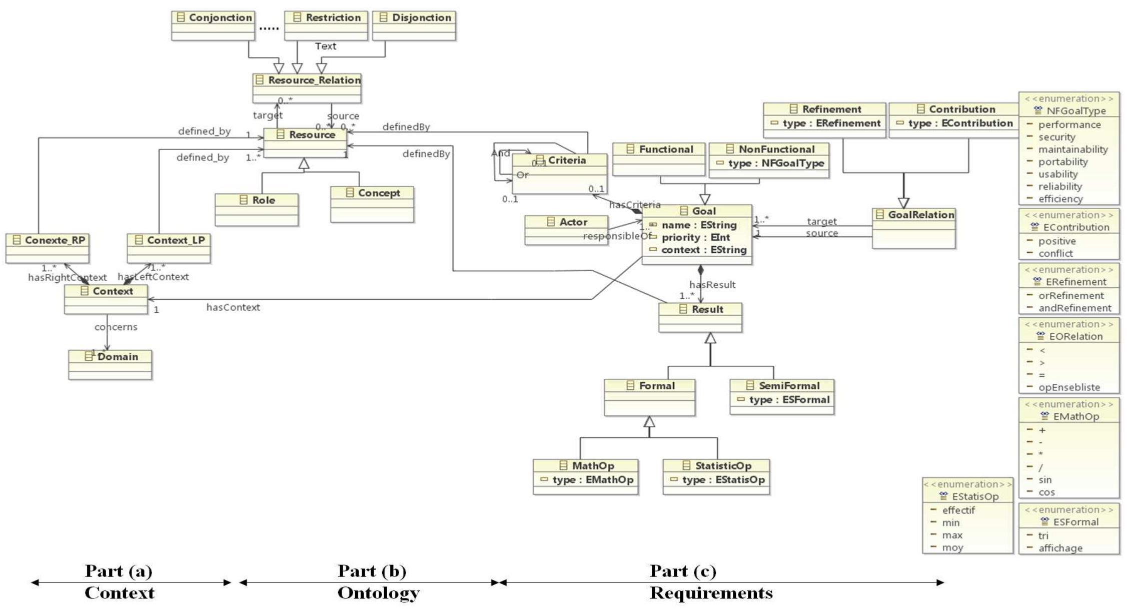


Fig. 1: Ontology model integrating context linked to the context model proposed

We extended the proposed formalization by considering the contextualizing and contextualized properties, as follows: $O :< C, R, Ref(C)$, $\mathbf{Ref}_{Context}$**, Mes (R)**, $formalism >$ such as:

$\mathbf{Ref}_{Context}$ is defined as a mapping function from the power set of R onto R : $CX2^R \rightarrow CXR$.

The properties defined in the right side of the mapping are the left side represent *contextualizing* properties, and the property defined in the right side of the function is the *contextualized* property. For the example given above, we would have: $Ref_{Context}$: $(MedicalUnit, MedicalUnitId), (Patient, IdPatient) \rightarrow (Time, Hospitalizationduration)$.

Mes (R) denotes a function $R \rightarrow Unit$ to define the unit of measurement of each role R. Units of measurement are necessary for formulas calculation that can be different from one context to another. Figure 1 illustrates the extension of the ontology model with the contextual model (part (a) and (b)). Figure 2 illustrates an example of the hospital ontology related to the context.

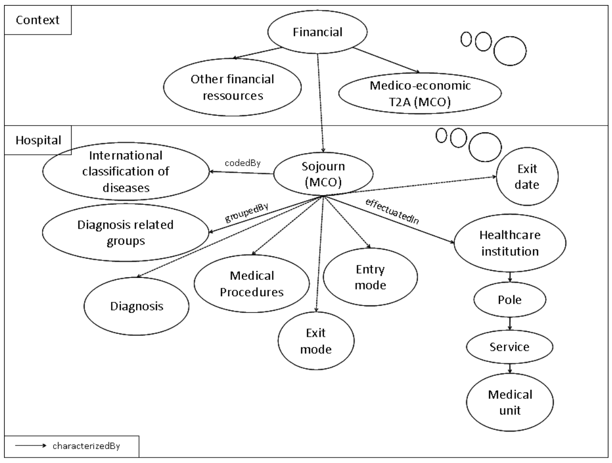


Fig. 2: Example of the hospital ontology related to the context

5.2 F_2: User requirements defined on multi-contextual ontologies

User requirements are expressed on the ontological level by the means of the goal oriented paradigm. Goal driven analysis [10] has been frequently used for $\mathcal{DW}$ development. It identifies goals and objectives that guide decisions of the organization at different levels. A goal is an objective that the system under consideration should achieve. Identifying goals of users is a crucial task for $\mathcal{DW}$ development. Indeed, the $\mathcal{DW}$ is at the core of a decisional application that needs to analyze the activity of an organization and where goals are important indicators of this activity. User's goals have a significant role in our method. They are used to identify the most relevant data to materialize in the $\mathcal{DW}$.

After analyzing works of goal-oriented literature, we proposed a Goal model considered as a pivot model since it combines three widespread goal-oriented approaches: KAOS, Tropos and iStar. The model is presented in Fig. 1 (Part (C)).

Let us take the following goal *Goal1*: "Reduce the number of unbilled records for the responsible of the medical information department (DIM)". The goal model is composed of main entity Goal described by some characteristics (name, context, priority). A goal is issued and achieved by some actors (DIM's responsible). A goal is characterized by two coordinates: (1) a *Result* to analyze (Reduce the number of T2A unbilled records) that can be quantified by given for-mal or semi-formal metrics measuring the satisfaction of the goal (number of un-coded records/total of records), and (2) some *Criteria* influencing this result (doctor) coding responsible. Two types of goals are identified: *functional* and *non-functional* goals. A non-functional requirement is defined as an attribute or constraint of the system (such as security, performance, flexibility, etc). Two types of relationships between goals are distinguished (reflexive relations): *AND/OR* relationships decomposing a general goal into sub-goals and *influence* relationships (positive, negative or ambiguous influence).

5.3 F_3: Definition of the integration framework for integrating sources containing contextual data

We define an integration framework $<\mathcal{G}, \mathcal{S}, \mathcal{M}>$ adapted to contextual data.

The global schema G: Schema $\mathcal{G}$ is represented by a Global Ontology (GO). As explained, the global ontology is formally defined to handle contextual information as follows $O :< C, R, Ref(C), Ref_{Context}, Mes(R), formalism >$. Note that the definition of the GO concerns only its TBox, which is usually assumed in data integration systems.

The sources S: The set of sources considered are semantic databases ($\mathcal{SDB}$s). Each $\mathcal{SDB}$ is defined by its local ontology (O_i) and its instances part (the ABOX). As, explained previously, the ontology model and its instances can be stored using different storage layouts. $\mathcal{SDB}$s may have different architectures. A $\mathcal{SDB}$ is formally defined as follows $< O_i, I, Pop_c, SL_{O_i}, SL_I, Ar >$ where:

- O_i: $< C, R, Ref, Ref_{context}, Mes(R), formalism >$ is the *ontology* model of the $\mathcal{SDB}$.
- I: presents the *instances* (the ABox) of the $\mathcal{SDB}$.
- $Pop_C : C \rightarrow 2^I$ is a *function* that relates each concept to its instances.
- SL_{O_i}: is the *Storage Layout* of the ontology model. We distinguish three main *relational* representations [1]: *vertical*, *binary* and *horizontal*. Vertical representation stores data in a unique table of three columns (subject, predicate, object) [27]. In a binary representation, classes and properties are stored in different tables [13]. Horizontal representation translates each class as a table having a column for each property of the class.
- SL_I: is the *Storage Layout* of the instances I. The storage layout used for instances can be the same layout used for storing the ontology, or a different one.
- Ar: is the *architecture* of the $\mathcal{SDB}$.

The mappings M: Mappings assertions relate a mappable element (MapElmG) of schema G (MapSchemaG) to a mappable element ($MapElmS$) of a source schema (MapSchemaS). These assertions can be defined at the intensional level (TBox) or at the extensional level ($ABox$). Different types of semantic relationships can be defined between mappable elements (Equivalence, Containment or Overlap). Discovering such mappings is related to the domain of schema and ontology matching/alignment, which is out of the scope of this paper. The mapping assertions are formally defined as follows M:< MapSchemaG, MapSchemaS, MapElmG, MapElmS, Interpretation, SemanticRelation >. This formalization is based on [21] meta-model:

- **MapSchemaG** and **MapSchemaS**: present respectively the *mappable schema* of the global and the local ontology.
- **MapElmG** and **MapElmS**: present respectively a *mappable element* of the global and the local ontology schema. This element can be a simple concept, instance or an expression (*Exp*) over the schema.
- **Interpretation**: presents the *Intentional* interpretation or *Extensional* interpretation of the mapping. In our study, the availability of global and local ontologies allows to define intentional mappings.
- **SemanticRelation**: three relationships are possible: *Equivalence*, *Containment* or *Overlap*. Equivalence states that the connected elements represent the same aspect of the real world. Containment states that the element in one schema represents a more specific aspect of the world than the element in the other schema. Overlap states that some objects described by an element in one schema may also be described by the connected element in the other schema.

5.4 F_4: Design method

We propose a method for designing a semantic $\mathcal{DW}$ covering the following steps: requirements analysis, conceptual design, logical design, ETL process, deployment and physical design. Fig. 3 illustrates these steps.

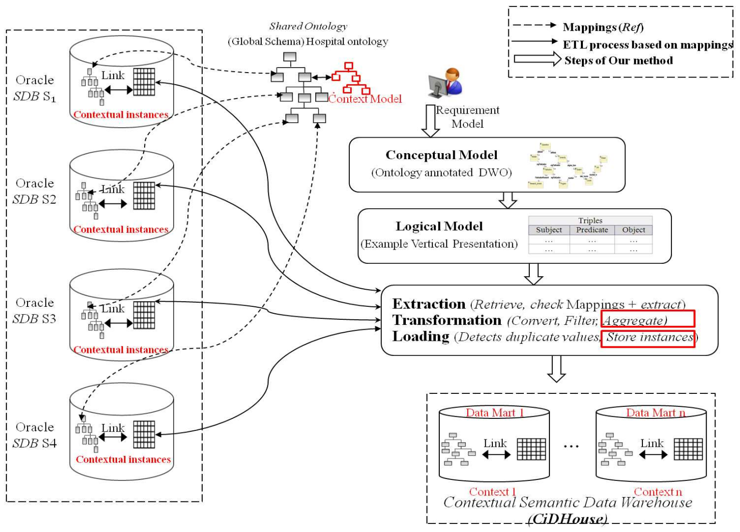


Fig. 3: Design method proposed.

Requirements analysis: This step allows the designer identifying the following: (1) the set of relevant properties used by the target application and (2) the set of treatments it should answer. The first set allows the construction of the *dictionary* containing the relevant concepts required for the application. As the ontology describe all concepts and properties of a given domain, a connection between requirement model and the ontology model is feasible. To do so, we define a connection between coordinates of each goal (*Result* and *Criteria*) and the resources (*concepts* and *roles*) of the GO (Fig. 1- Part B and C). This allows the designer to choose the most relevant ontological concepts to express user's goals. Knowing that the GO is linked to the data sources, these concepts chosen to express goals inform the designer about the most relevant data to store in the $\mathcal{DW}$ model.

Conceptual Design: A $\mathcal{DW}$ ontology (DWO) (that can be viewed as a conceptual abstraction of the $\mathcal{DW}$) is defined from the global ontology (GO) by extracting all concepts and properties used by user goals. Three scenarios materialize this definition:

1. $DWO = GO$: the GO corresponds exactly to users' requirements,
2. $DWO \subset GO$: the DWO is extracted from the GO,
3. $DWO \supset GO$: the GO does not fulfill all users' requirements.

The designer may extend the DWO by adding new concepts and properties in the case, where the GO does not satisfy all her/his requirements. The concepts belonging to DWO and do not reference any concept of sources are annotated and are set by *null values* in the target warehouse.

The multidimensional role of concepts and properties are then discovered and stored as ontological annotations. We propose the algorithm 1 for multidimensional annotations.

Figure 4 presents the multidimensional model generated after applying the 1 on the ontology hospital and the set of goals collected from users.

```
begin
    for Each goal G do
        Each concept (resp. role) used as a result of $G$ is a fact (resp. measure) candidate;
        Each concept (resp. role) used as a criterion of $G$ is a dimension (resp. dimension attribute)
        candidate;
        Criteria of goals influencing G are dimension candidates of the measure identified for $G$;
        Concepts of measures are facts candidates;
        Concepts of dimension attributes are dimension candidates;
        if fact concept $F$ is linked to a dimension by (1,n) relationship then
            keep the two classes in the model
        else
            Reject the dimension class;
        end
        Hierarchies between dimensions are constructed by looking for (1,n) relationships between
        classes identified as dimensions (for each fact);
    end
    Generalization (is-a) relationships existing in the ontology between facts or between dimensions
    are added in the model.;
end
```

Algorithm 1: Multidimensional annotations

Logical Design: The logical $\mathcal{DW}$ model is generated by translating the *DWO* to a relational model (other data models can be chosen). Several works in the literature proposed methods for translating ontologies described in a given formalism (PLIB, OWL, RDF) to a relational or an object-relational representations [8].

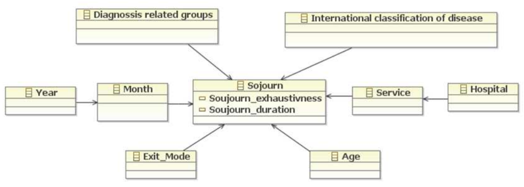


Fig. 4: Multidimensional schema (snowflake) of the case study

ETL process: The goal of the ETL process is to populate the target $\mathcal{DW}$ schema obtained in the previous step, by data of sources. [22] defined ten generic operators typically encountered in an ETL process, which are: EXTRACT (S,C), RETRIEVE(S,C), MERGE(S,I), UNION (C,C'), JOIN (C, C'), STORE(S,C, I), DD(I), FILTER(S,C,C'), CONVERT(C,C') and AGGREGATE (F, C, C').
These operators have to be leveraged to deal with the semantic aspects of sources. Therefore, we propose an Algorithm 2 for populating the *DWO* schema. The algorithm is based on the generic conceptual ETL operators as presented above. They can then be instantiated according to one of the storage layouts: vertical, binary, horizontal. Each operator will correspond to a defined query. Sparql is the standard query language used for querying ontologies. We translated each operator to a Sparql query. For instance, the EXTRACT operator is translated as follows:
Select ?Instance#
Where {*?Instance# rdf:type nameSpace:Class. ?Instance NameSpace:DataProperty value_condition*}.
The above statement shows a SPARQL query used on a vertical storage layout (triples). The

query selects all triples from a source, and then inserts them into a staging table of oracle SDB using a SQL query.

In order to obtain a contextual $\mathcal{DW}$, we adapted the following operators to manage the contextual instances: the AGGREGATE operator and the STORE operator. We adapted the AGGREGATE operator so that the aggregate function is applied according to the context defined on the instance. Different contexts can use different calculation formulas. The aggregate functions can thus be distinct from one context to another. The AGGREGATE operator is adapted as follows:
Select (Aggregate_Function(?Instance) AS ?Aggregate_Function)
Where {?Instance rdf:type namespace:Class.
namespace:DatatypeProperty rdfs:domain namespace:Class.
***namespace:DatatypeProperty rdfs:label ContextID**}*
Group By ?Instance

Example 1. For example, assuming that the namespace of Hospital Ontology is

```
PREFIX OntoHospital: < http://www.semanticweb.org/p091417/ontologies/2013/2/untitled-ontology-62>
```

The following query applies the COUNT function on the Patient instances according to the service context.
Select number of patients per context: Select (count(?Instance) AS ?patients)
Where {?Instance rdf:type OntoHospital:Patient.
OntoHospital:DatatypeProperty rdfs:domain OntoHospital:Class.
OntoHospital:DatatypeProperty rdfs:label "Cardiology"}
Group By ?Instance

The STORE operator is adapted so that it loads instances corresponding to a class in the target data store according to the context of source instances. The obtained data store is a contextualized data mart storing instances of one context. The following statement shows a SPARQL query selecting all triples from a source according to their context then inserts them into a staging table of oracle SDB using a SQL query:
SELECT ?Instances ?ContextID
Where {?Instances rdf:type namespace:Class.
?DatatypeProperty rdfs:domain namespace: Class.
***?DatatypeProperty rdfs:label ?ContextID**.*
?DatatypeProperty rdfs:label ContextID }
Insert into staging_table Values (id, SDO_RDF_TRIPLE_S (?Instances, rdf:type , namespace:Class));

Example 2. For example, the following queries are used to select patients per context and load them into a staging table of oracle SDB:
SELECT ?Instances ?ContextID
Where {?Instances rdf:type OntoHospital:Patient.
?DatatypeProperty rdfs:domain OntoHospital: Patient.
?DatatypeProperty rdfs:label ?ContextID.
?DatatypeProperty rdfs:label ContextID }
Insert into staging_table Values (id, SDO_RDF_TRIPLE_S (?Instances, rdf:type , OntoHospital:Patient));

Based on the framework <GO,$\mathcal{SDB}$,M >, the integration process depends on the *semantics of mappings* (*SemanticRelation*) between *GO* and local ontologies ($\mathcal{SDB}$), where four semantics mappings are identified: (1) *Equivalent* ($C_{GO} \equiv C_{SDB}$) and (2) *Containment sound* ($C_{GO} \supset C_{SDB}$): where no transformation is needed. Instances are extracted from sources, merged, united or joined then loaded in the target data store. (3) *Containment complete* ($C_{GO} \subset C_{SDB}$): where source instances satisfy only a subset of the constraints required by *GO* classes, some instances need to be transformed (converted, filtered and aggregated) then merged, unified or joined and finally loaded to the target data store. (4) *Overlap* mappings: where we need to identify the constraints required by GO classes and not applied to the source classes. This case is then treated same as the Containment (Complete) scenario. Algorithm 2 (presented in a previous paper [4]) depicts the ETL process based on these four scenarios.

Deployment and Physical Design The target $\mathcal{DW}$ model defined by our proposal is populated according to a given DBMS. We validated our proposal using Oracle DBMS. An Oracle $\mathcal{SDB}$ is used to store the target CiDHouse model and the ontology defining its semantics.

Oracle has incorporated support for languages RDF and OWL in its system to enable its customers to benefit from a management platform for semantic data. Oracle has defined two subclasses of DLs: *OWLSIF* and a richer fragment *OWLPrime*. We use *OWLPrime* fragment which offers the following constructors[5]: *rdfs:domain, rdfs:range, rdfs:subClassOf, rdfs:subPropertyOf, owl:equivalentClass, owl:equivalentProperty, owl:sameAs, owl:inverseOf, owl: TransitiveProperty, owl:SymmetricProperty, owl:FunctionalProperty, owl: InverseFunctionalProperty*. Note that *OWLPrime* limits the expressive power of DL formalism in order to ensure decidable query answering.

We first instantiated the integration framework $< G, S, M >$ using the hospital ontology and Oracle semantic sources containing contextual instances, as follows:

The global schema G: The global schema is represented by the Hospital ontology, and is formalized as follows:

$\mathbf{O}_{Oracle}$: <Classes C, Properties P (Datatype Property and Object Property), Ref:(Operator, Expressions), Ref$_{context}$, Mes(R), OWLPrime>

For this case study, Ref is the function that gives the expression (or definitions) of classes and properties using operators available in *OWLPrime* (rdfs:subClassOf, owl:equivalentClass, rdfs:subPropertyOf, owl:equivalentProperty). *Expression* is an expression over classes and properties using *OWLPrime* constructors described above.

The local sources S: Each source is constructed from the hospital ontology, by using a set of defined mappings. Each local source S_i is instantiated for Oracle $\mathcal{SDB}$ as follow:

S_i: $< O_{Oracle}$, Individuals (triples), Pop is given in tables RDF_link$ and RDF_values$, Vertical, Vertical, type I>. Vertical storage is a relational schema composed of one table of triples (subject, predicate, object). For example: *(Patient, type, Class)* for the ontology storage and *(Patient#1, type, Student)* and *(Patient#1, affected, Service#1)* for the instances storage.

The mappings: Mapping assertions between global and local schema are instantiated for Oracle as follows:

Mapping M:$< O_{Oracle}$ of each source, O_{Oracle} of the global schema, Expression over G, Class of a source S, Intensional interpretation, (*Equivalent, Containment or Overlap* (owl:SubClassOf and owl:equivalentClass in OWLPrime))>.

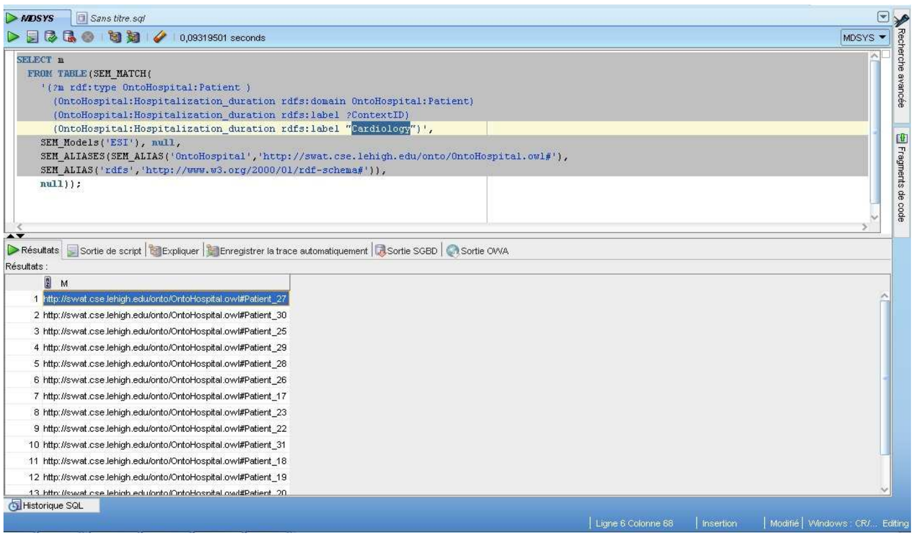
We then applied the ETL algorithm proposed, based on the ETL operator adapted to deal with the contextual instances. The result of the integration process is a $\mathcal{DW}$ whose schema corresponds to the classes and properties of the hospital ontology IO, populated by instances selected from Oracle $\mathcal{SDB}$s. Figure 5 illustrates the results (instances) of the defined CiDHouse.

6 Conclusion

Real life applications are usually shared by many users from heterogeneous domains. Consequently, data managed by these applications contain contextual information. Managing contextual data is a challenge issue especially for decisional applications. We proposed in this paper an design method for $\mathcal{DW}$ applications, that deals with contextual information from the first phase of $\mathcal{DW}$ design. We proposed a formal definition of domain ontologies enriched with context information. The design

[5] http://www.w3.org/2007/OWL/wiki/OracleOwlPrime

```
begin
    Input:
    $DWO$: $\mathcal{DW}$ Ontology (Schema) and Si: Local Source ($\mathcal{SDB}$)
    Output: $DWO$ populated (schema + instances)
    for Each C : Class of ontology DWO do
        $I_{DWO} = \phi$
        for Each source Si do
            if $Cs \equiv C$  /* instances in Si satisfy all constraints imposed by $DWO$*/
            then
                C'= IdentifyClasse (Si, C) /*identify class from Si*/
            else
                if $Cs \subset C$  /*Instances in Si satisfy all constraints imposed by $DWO$,
                plus additional ones */ then
                    C'= IdentifyClasse (Si, C) /*identify class from Si*/
                else
                    if Cs$\supset$ C Or Overlap mappings /* Instances satisfy only a subset of
                    constraints imposed by $DWO$*/ then
                        if format(C) $\neq$ format(Cs) then
                            Cconv= CONVERT (C, Cs) /*identify the constraint of format
                            conversion from the source to the target $DWO$*/
                        end
                        if C represent aggregation constraint then
                            Caggr= AGGREGATE (F, C, Cs) /*identify the constraint of
                            aggregation defined by F*/
                        end
                        if C represents filter constraint then
                            Cfilt= FILTER (Si, C, Cs) /*identify the filter constraint
                            defined in the target $DWO$*/
                        end
                        C'= ClasseTransformed (Si, C, Cconv, Caggr, Cfilt) /* Associate to the
                        class C' the constraint of conversion, aggregation or filtering
                        defined by Cconv, Caggr and Cfilt*/
                    end
                end
            end
            $I_{si}$= RETRIEVE (Si, C') /*Retrieve instances of C' and applying constraints
            of conversion, aggregation or filtering if necessary*/
            if more than one instance are identified in the same source then
                $I_{DWO}$= MERGE ($I_{DWO}$, $I_{si}$) /*Merge instances of Si*/
            end
            if classes have the same super class then
                $I_{DWO}$= UNION ($I_{DWO}$, $I_{si}$) /*Unites instances incoming from different
                sources*/
            else
                if  classes are related by same property then
                    $I_{DWO}$= JOIN ($I_{DWO}$, $I_{si}$) /* Join incoming instances*/
                end
            end
            if Source contain instances more than needed then
                $I_{DWO}$= EXTRACT ($I_{DWO}$, $I_{si}$) /* Extract appropriate portion of
                instances*/
            end
        end
        STORE(DWO,C, DD($I_{DWO}$)) /*Detects duplicate values of instances and load
        them in DWO*/
    end
end
```
Algorithm 2: The population of the $\mathcal{DW}$ by the means of ontological ETL operators



```
SELECT m
  FROM TABLE(SEM_MATCH(
    '(?m rdf:type OntoHospital:Patient )
      (OntoHospital:Hospitalization_duration rdfs:domain OntoHospital:Patient)
      (OntoHospital:Hospitalization_duration rdfs:label ?ContextID)
      (OntoHospital:Hospitalization_duration rdfs:label "Cardiology")',
    SEM_Models('ESI'), null,
    SEM_ALIASES(SEM_ALIAS('OntoHospital','http://swat.cse.lehigh.edu/onto/OntoHospital.owl#'),
    SEM_ALIAS('rdfs','http://www.w3.org/2000/01/rdf-schema#')),
    null));
```

```
0,09319501 seconds

Résultats :
     M
 1 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_27
 2 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_30
 3 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_25
 4 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_29
 5 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_28
 6 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_26
 7 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_17
 8 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_23
 9 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_22
10 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_31
11 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_18
12 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_19
13 http://swat.cse.lehigh.edu/onto/OntoHospital.owl#Patient_20
```

Fig. 5: Results of the integration process in the CiDHouse

method follows five design steps: requirements definition, conceptual design, logical design and ETL process. The ETL step is processed through an algorithm based on ETL operators. We adapted a set of these operators in order to manage the contextual data. The method is validated using a practical case study from the medical domain. Our design approach assists end users during the exploitation phase of the warehouse.

An interesting issue that has to be considered is the generation of data marts based on contexts of end users.

References

1. D. Abadi, A. Marcus, S. Madden, and K. Hollenbach. Sw-store: a vertically partitioned dbms for semantic web data management. *VLDB Journal*, 18(2):385–406, 2009.
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.
3. L. Bellatreche and al. Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases. *Computers in Industry Journal Elsevier*, 57(8-9):711–724, 2006.
4. L. Bellatreche, S. Khouri, and N. Berkani. Semantic data warehouse design: From etl to deployment a la carte. In *To Appear in the 18th International Conference on Database Systems for Advanced Applications (DASFAA)*, 2013.
5. L. Bellatreche, S. Khouri, I. Boukhari, and R. Bouchakri. Using ontologies and requirements for constructing and optimizing data warehouses. In *MIPRO, 2012 Proceedings of the 35th International Convention*, pages 1568–1573, 2012.
6. D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, and M. Vincini. Information integration: The momis project demonstration. *VLDB Journal*, page 611614, 2000.
7. S. Chaudhuri. How different is big data? In *ICDE*, page 5, 2012.
8. A. Gali, C. Chen, K. Claypool, and R. Uceda-Sosa. From ontology to relational databases. *ER Workshops*, pages 278–289, 2004.
9. I. Garrigós, J. Pardillo, J.-N. Mazón, and J. Trujillo. A conceptual modeling approach for olap personalization. In *Conceptual Modeling-ER 2009*, pages 401–414. Springer, 2009.
10. P. Giorgini, S. Rizzi, and M. Garzetti. Goal-oriented requirement analysis for data warehouse design. *DOLAP'05*, pages 47–56, 2005.

11. C. Goh, S. Bressan, E. Madnick, and M. Siegel. Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems*, 17(3):270–293, 1999.
12. V. Kashyap and A. P. Sheth. Semantic and schematic similarities between database objects: A context-based approach. *VLDB Journal*, 5(4):276–304, 1996.
13. J. Lu, L. Ma, L. Zhang, J. S. Brunner, C. Wang, Y. Pan, and Y. Yu. Sor: A practical system for ontology storage, reasoning and search. *In VLDB*, pages 1402–1405, 2007.
14. E. Mena, A. Illarramendi, V. Kashyap, and A. P. Sheth. Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Distributed and Parallel Databases*, 8(2):223–271, 2000.
15. B. Neumayr, S. Anderlik, and M. Schrefl. Towards ontology-based olap: datalog-based reasoning over multidimensional ontologies. In *Proceedings of the fifteenth international workshop on Data warehousing and OLAP*, DOLAP '12, pages 41–48, New York, NY, USA, 2012. ACM.
16. J. M. Pérez, R. Berlanga, M. J. Aramburu, and T. B. Pedersen. A relevance-extended multi-dimensional model for a data warehouse contextualized with documents. In *Proceedings of the 8th ACM international workshop on Data warehousing and OLAP*, DOLAP '05, pages 19–28, New York, NY, USA, 2005. ACM.
17. G. Pierra. Context-explication in conceptual ontologies: The plib approach. In *in Proceedings of 10th ISPE International Conference on Concurrent Engineering: Research and Applications (CE03) : Special Track on Data Integration in Engineering*, pages 243–254, 2003.
18. G. Pierra. Context representation in domain ontologies and its use for semantic integration of data. *Journal of Data Semantics (JoDS)*, 10:174–211, 2008.
19. Y. Pitarch, C. Favre, A. Laurent, and P. Poncelet. Enhancing flexibility and expressivity of contextual hierarchies. In *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, pages 1–8, 2012.
20. E. Pitoura, K. Stefanidis, and P. Vassiliadis. Contextual database preferences. *IEEE Data Eng. Bull.*, 34(2):19–26, 2011.
21. B. S., H. P., S. L., and S. H. Formal and conceptual comparison of ontology mapping languages. In *Modular Ontologies*, pages 267–291. Springer-Verlag, Berlin, Heidelberg, 2009.
22. D. Skoutas and A. Simitsis. Ontology-based conceptual design of etl processes for both structured and semi-structured data. *Int. J. Semantic Web Inf. Syst.*, 3(4):1–24, 2007.
23. K. Stefanidis, N. Shabib, K. Nørvåg, and J. Krogstie. Contextual recommendations for groups. In *Advances in Conceptual Modeling*, pages 89–97. Springer, 2012.
24. P. Vajirkar, S. Singh, and Y. Lee. Context-aware data mining framework for wireless medical application. In *DEXA*, pages 381–391, 2003.
25. V. T. V. U. S. H. S. G. N. H. Wache, H. and S. Hiibner. Ontology-based integration of information - a survey of existing approaches. In *OIS*, pages 108–117, 2001.
26. X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung. Ontology based context modeling and reasoning using owl. *In Proceeding of PERCOMW '04*, pages 18–22, 2004.
27. Z. Wu, G. Eadon, S. Das, E. Chong, V. Kolovski, M. Annamalai, and J. Srinivasan. Implementing an inference engine for rdfs/owl constructs and user-defined rules in oracle. In *ICDE*, pages 1239–1248, 2008.