Contexts as Explicit Parameterization of Ontology Driven Methods

Riccardo Albertoni, Elena Camossi, Monica De Martino, Franca Giannini, Marina Monti

CNR-IMATI,
Via De Marini, 6 - Torre di Francia - 16149 Genova, Italy
{albertoni,camossi,demartino,giannini,monti}@ge.imati.cnr.it,

Abstract. In the recent Computer Science literature, contexts have been proposed mainly to formalize context dependent background knowledge. In this paper we discuss the importance of the application of contexts as explicit parameterization of methods exploiting the knowledge encoded in ontologies. We propose a context formalization suitable to improve the flexibility of ontology driven methods like semantic similarity and granularity. Herein, we detail in particular the context parameterization for semantic granularity. The user scenario pertaining to the exploitation of a repository for industrial design product models is discussed to illustrate the proposed formalization.

1 Introduction

In recent literature, ontologies are adopted to represent semantic metadata characterizing different kinds of information resources [1]. Semantic metadata are costly to be compiled but result to be precious whenever the resources have to be selected for different applications. Thus, we consider the full exploitation of ontologies as a primary research issue. In this perspective, it becomes mandatory to taken into account contexts, which enable to bind the background domain knowledge encoded in ontologies to a local point of view [2].

The representation of context is an emerging issue, which is discussed in several fields like Cognitive and Computer Science [3]. In particular, in Information Retrieval and related areas, contexts are employed to represent information user needs [4,5,7,8], often through the use of ontologies. In the field of Knowledge Representation for the Semantic Web, proposals to support local (contextual) representation of the background knowledge have been presented [2,9]. Moreover, in Computer Science the concept of context is often related to the notion of *view* [6,10,11,12]. Despite of the big amount of proposals, unfortunately current reasoning techniques still lack of context-dependent ontology driven methods to enhance the sift of ontology instances, e.g., techniques to organize resources at different levels of abstraction and to assess their similarity.

In this paper, we provide a formalization of application context to parameterize different ontology driven methods. The proposal extends the definition of application context for semantic similarity introduced in [13]. In particular, we consider as an example of further ontology driven method to be parameterized, the semantic granularity

described in [14]. Granularities enable the browsing of information resources according to different levels of detail, i.e., granularities. They have been already studied in the area of Information Systems, in particular for the spatio-temporal domain [15]. However, in this field, granularities are static and embedded in the data model or in the database schema. Some attempts to define semantic granularities have been made with respect to terminologies by Fonseca et al. [16]. In [17], spatial and temporal granularities are also employed to constrain the objects of interest in a context. Our work extends the formalism of granularity presented in [14] in order to parameterize its application according to the notion of application context.

As a first validation of the context formalization we propose, we discuss a typical product design application scenario illustrating different application contexts in which a designer browses a repository of existing models organized with respect to their functionalities.

The paper is organized as follows. In Section 2 we introduce the framework for the definition of ontology driven methods. In Section 3 we formalize the application contexts. This formalization is employed to extend the extraction of semantic granularity according to contexts in Section 4. Section 5 concludes the paper outlining some future research directions.

2 A Layered Framework for Ontology Driven Methods

Ontology driven methods are contextualized relying on an ontology model and a layered framework inspired by Ehrig et al. [18]. The ontology model provides the expressiveness of the ontologies defined according to the framework. The framework is structured in terms of *data*, *ontology* and *context* layers plus the *domain knowledge* layer which spans all the others. Our formalization extends the ontology model and the layered framework to support ontology driven methods explicitly parameterized according to a context. Hereafter, ontologies with data types are considered as the ontology model. An ontology with data types is defined as a structure $O := (C, T, \leq_C, R, A, \sigma_R, \sigma_A, \leq_R, \leq_A, I, V, l_C, l_R, l_A)$ where the C, T, R, A, I, V are disjointed sets, respectively, of classes, data types, binary relations, attributes, instances and data values; $\leq_C$, $\leq_R$, $\leq_A$ define, respectively, the class, relation and attribute hierarchies; $\sigma_R : R \to C \times C$ and $\sigma_A : A \to C \times T$ define the signature for each relation and attribute; and $l_C : C \to 2^I$, $l_T : T \to 2^V$, $l_R : R \to 2^{I \times I}$, $l_A : A \to 2^{I \times V}$ are the functions for the instantiation of classes, data type values, relations and attributes, respectively.

Additionally, we have defined also the functions that retrieve the attributes, the relations and the concepts reachable in a navigational path. In particular, we define[1]:

- δ_a: $C \to 2^A$, where $\delta_a(c) = \{a\text{: } A \mid \exists t \in T, \sigma_A(a) = (c, t)\}$ retrieves the attributes of c;
- δ_r: $C \to 2^R$, where $\delta_r(c) = \{r\text{: } R \mid \exists c' \in C, \sigma_R(r) = (c, c')\}$ retrieves the relations of c;
- δ_a: $R \to 2^A$, where $\delta_a(r) = \{a\text{: } A \mid \exists c, c' \in C \exists t \in T, \sigma_R(r) = (c, c') \wedge \sigma_A(a) = (c', t)\}$ retrieves the set of attributes of the classes reachable by the relation $r \in R$;

[1] Note that each function name corresponds to a couple of functions which differ in the kind of parameter they take.

- δ_c: $R \rightarrow 2^C$, where $\delta_c(r) = \{c'\text{: } C \mid \exists c \in C, \sigma_R(r) = (c, c')\}$ retrieves the set of concepts reachable by the relation $r \in R$;
- δ_c: $C \rightarrow 2^C$, where $\delta_c(c) = \{c'\text{: } C \mid \exists r \in \delta_r(c); \sigma_R(r) = (c, c')\}$ retrieves the set of concepts related to $c \in C$ through a relation in R;
- δ_r: $R \rightarrow 2^R$, where $\delta_r(r) = \{r'\text{: } R \mid \exists c \in C, \exists c' \in \delta_c(r); \sigma_R(r') = (c', c)\}$ retrieves the set of relations of the concepts reachable through the relation $r \in R$;
- $\delta_{r^{-1}}$: $C \rightarrow 2^R$, where $\delta_{r^{-1}}(c) = \{r\text{: } R \mid \exists c' \in C, \sigma_R(r) = (c', c)\}$ retrieves the set of relations that reach $c \in C$;
- $\delta_{r^{-1}}$: $R \rightarrow 2^R$, where $\delta_{r^{-1}}(r) = \{r'\text{: } R \mid r' \neq r,\ \exists c \in C, \exists c' \in \delta_c(r), \sigma_R(r') = (c, c')\}$ retrieves the set of relations which differ from r and reaches the concepts reachable through the relation $r \in R$.

These functions will be extensively employed in the formalization of the ontology and the context layers. In our proposal, the *data layer* provides different *functions* on data type values (e.g., functions which measure the similarity of values of simple or complex data types, statistical and user defined functions). All these functions are explicitly plugged in by considering the data type on which they can be applied. The *ontology layer* provides the mechanism for processing a set of ontology driven methods by considering the way ontology's entities are related. For each ontology driven method, ontology layer provides the implementation of the *operations* (e.g., intersection, count) which can be recalled in the application context. The *context layer* provides the *application contexts*, i.e., the criteria for the computation of ontology driven methods considering how ontology entities are used in external contexts. Each application context specifies the attributes and the relations to be considered as well as operations and functions to be applied on them. Each method available in the *ontology layer* behaves differently according to the specific application context.

In this paper, we consider the semantic similarity among ontology instances described in [13] and an extension of the semantic granularity presented in [14] as an examples of context-dependent ontology driven methods. The framework can be extended with further ontology driven methods.

3 Formalization of Application Contexts

In this section, we formalize the restrictions that an application context must adhere to. The formalization presented herein extends [13]. The formalization relies on the concepts of *sequence of elements* and *path of recursion*.

Given a set X, a *sequence s of elements* in X with length n is defined by the function $s : (i) \rightarrow X$, with i a set of index such that $i \in N^+$. The sequence can be represented by the list of functional values $[s(1), .., s(n)]$. $S_X^n = \{s \mid s : (i) \rightarrow X\}$ is the set of sequences of X having length n, and $\circ : S_X^n \times S_Y^m \rightarrow S_{X \cup Y}^{n+m}$ is the operator to concatenate two sequences. A *path of recursion* tracks the recursion during the assessment of an ontology driven method, and represents the navigation path in the ontology to collect the information of interest. More formally, a path of recursion p with length n is a *sequence* whose first element is a class c and whose other elements are relations starting from or arriving at c or one of the classes that are reached by the previous relations in the path of recursion, such that given $p \in S_{C \cup R}^n$, $p(1) \in C \wedge \forall j \in [2, n]\ p(j) \in R \wedge (p(j) \in$

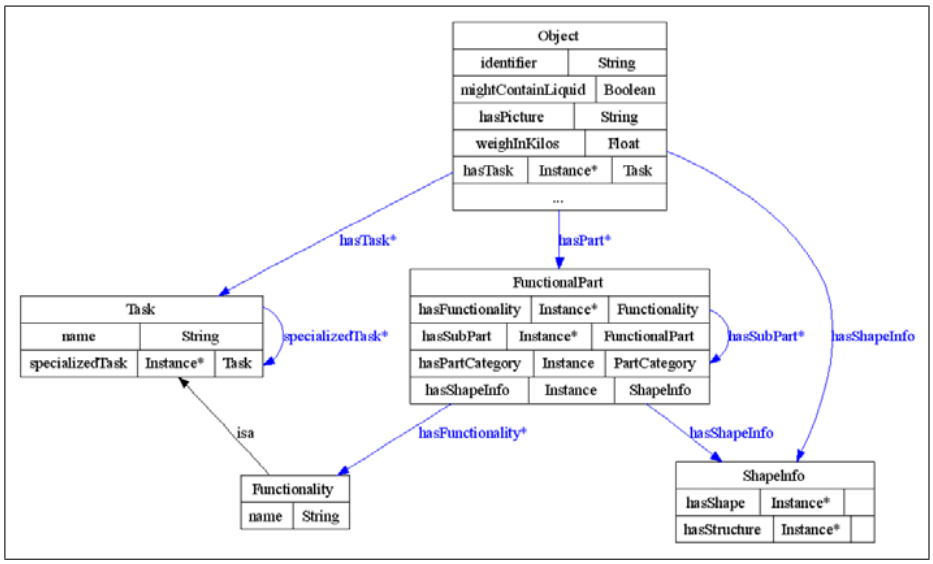

Fig. 1. Industrial design product ontology

$\delta_r(p(j-1)) \vee p(j) \in \delta_{r^{-1}}(p(j-1)))$. P^i is the set of all paths of recursion with length i, whereas P is the set of all paths of recursion $P = \bigcup_{i \in N} P^i$.

Given for example the ontology schema of Figure 1, which reports information about product models, such as its subdivision into components (e.g., handle, body, filter), its shape representation (e.g., mesh, graph, etc.), the tasks an object is designed to accomplish with (e.g., to cook, to boil), the functionalities provided by its parts (e.g, to pour, to contain, to sieve), an example of path of recursion of length 3 is: [*Object.hasPart.hasFunctionality*]. This path of recursion starts from class `Object`, reaches class `FunctionalPart` through relation `hasPart`, and finally reaches class `Functionality` through relation `hasFunctionality`. We say that the classes `Object`, `FunctionalPart`, `Functionality` are reachable through this path of recursion.

In [13] the *application context (AC)* function is defined inductively according to the length of the path of recursion. It yields the set of attributes and relations as well as the operations to be used computing the ontology driven methods. In its original form the application context had a single set of operations applicable to compare relation and attribute values during the similarity assessment. These operations in the case of attribute values could indirectly recall the functions provided by the data layer. We extend the previous formalisms with functions that can be applied to the same datatype according to the attribute semantics when computing ontology driven methods. For example, for datatype `integer`, functions which can be considered are: `sum`, `average`, `minimum`, `maximum`.

Moreover, we consider also different sets of operations, defined in the ontology layer, which can be considered in the context layer according to the ontology driven

method to apply. For example, for semantic similarity we considered *Count*, *Inter*, *Simil* to evaluate respectively the cardinality, the intersection, the similarity of instances or values associated to the considered relation or attribute. In the case of semantic granularity we include different forms of count operations: *Count*, to evaluate the cardinality of a set of instances; *WCount*, to evaluate a weighted count of instances, according to the cardinality of related attributes or relations; *InvCount*, to evaluate the inverse cardinality of a set of instances, (i.e., set with less instances has more importance than set with greater cardinality).

The application context is formally defined as follows.

Definition 1 (Application Context AC) *Given the set P of paths of recursion, n the number of ontology driven methods available, $i \in [1, n]$, L_i the set of operations provided by the ontology layer for the i-th method, G_i the set of datatype functions available in the data layer for the i^{th} method, the application context for the i^{th} ontology driven method is defined by the partial function AC_i: $P \rightarrow 2^{A\times(L_i\cup G_i)} \times 2^{R\times L_i}$.* △

Note that each application context AC_i is characterized by the operators $AC_{i,A} : P \rightarrow 2^{A\times(L_i\cup G_i)}$ and $AC_{i,R} : P \rightarrow 2^{R\times L_i}$, which yield respectively the context AC_i related to the attributes and to the relations. For each path of recursion p, $AC_i(p) = (AC_{i,A}(p), AC_{i,R}(p))$ represents the portion of application context related to the path of recursion p for the i^{th} ontology driven method. In particular, $AC_{i,A}(p) = \{(a_1, opg_1), (a_2, opg_2), ..., (a_l, opg_l)\}$ and $AC_{i,R}(p) = \{(r_1, op_1), (r_2, op_2), ..., (r_k, op_k)\}$, with $opg_j \in L_i \cup G_i$, and $op_j \in L_i$, are, respectively, the attributes and the relations with the corresponding operations to be used computing the i^{th} ontology driven method on the path of recursion p.

Example 1. Given the ontology schema in Figure 1, an example of application context defined for the i^{th} ontology driven method starting from the path of recursion $[Task]$ is:

$$[Task] \overset{AC_i}{\rightarrow} (\{\}, \{(hasTask^{-1}, Count)\})$$

where the first part pertaining to the attributes is empty because no attributes are considered in this application context, whereas in the second part $hasTask^{-1}$ is the inverse relation of *hasTask*, and *Count* is the operation to be employed. This portion of application context requires, for the evaluation of semantic granularity, to evaluate the cardinality of the instances of `Object`, which are related to instances of `Task` through $\texttt{hasTask}^{-1}$. □

4 Context Dependent Semantic Granularities

In this section we provide a context dependent extension of semantic granularity as presented in [14]. Specifically, we extend this ontology driven method to extract semantic granularities according to the application context. The granularity system is built with respect to an ontology O representing the information resources that are described by a set of *qualities* Q and the relations among them. Information resources, which are going to be browsed, are instances of a class S. The set of qualities Q are represented by ontology classes organized in a hierarchy $\prec_Q$ induced by relations IS-A (i.e., $\leq_C$ in the ontology model) and Part-Whole (i.e., a relation $r \in R$ which relates objects and their

parts, as specified in [14]). We suppose that a class Q^T exists such that, for each quality Q, $Q \prec_Q Q^T$; moreover, each $Q \in \mathcal{Q}$ has at least one direct instance.

The user is expected to access the repository by using increasing levels of detail, i.e., granularities, which correspond to increasing detailed qualities, according to which the resources are grouped. Semantic granularities are defined dynamically, according to both the data model, represented by the ontology schema, and the data, given by ontology instances. The method follows a two-phase process. First, the *quality filtering* phase evaluates each quality with respect to its ability of abstracting information resources. Then, the *granularity building* phase distributes the qualities that have been returned by the filtering phase, namely the granules, among different granularities according to $\prec_Q$. This phase returns the set of granularities to employ for the repository navigation. Since not all the qualities in the hierarchy will be evaluated as good abstractors by quality filtering, the browsing of the information resources according semantic granularities will differ from the browsing driven by IS-A and Part-Whole.

The evaluation of the abstraction capability of a quality Q is obtained working out the ratio R_Q, which takes into account both the relations in which the quality is involved and the resources in $\mathcal{S}$ defined for it: the more R_Q is close to zero the more Q is a good abstractor. R_Q is defined in term of the relevance of the resources associated to the quality Q, which is denoted by s^Q, and the relevance of the resources associated to Q and its sub-qualities Q' with respect to $\prec_Q$, i.e., $Q' \prec_Q Q$, denoted by s^{Q^*}.

Definition 2 (Abstraction Capability of Quality Q) *Given an ontology $\mathcal{O}$, representing the class of information resources $\mathcal{S}$ described with respect to the set of qualities $\mathcal{Q}$, and the partial order $\prec_Q$ on $\mathcal{Q}$, the abstraction capability of a quality $Q \in \mathcal{Q}$ with respect to the hierarchy $\prec_Q$ is defined as:*

$$R_Q = \frac{max_{\{Q' | Q' \prec_Q Q\}} |s^{Q'^*}|}{\sum_{\{Q' | Q' \prec_Q Q\}} |s^{Q'^*}| + s^Q} \tag{1}$$

△

According to the aim induced by the application context, *quality filtering* can be evaluated taking into account different attributes and relations, as well as the application of operations on them. In particular, the evaluation of s^Q according to the application context AC is evaluated as the sum of the relevance of the instances q of Q. Similarly, the evaluation of s^{Q^*} takes into account also the instances of the "sub-qualities" of Q according to $\prec_Q$. The following definition formalizes the evaluation of s^Q and s^{Q^*} according to the application context AC.

Definition 3 (Context Dependent s^Q and s^{Q^*}) *Considering two qualities Q and Q' in $\mathcal{Q}$, classes in $\mathcal{O}$, Q^T the most generic quality in the qualities's hierarchy, p the starting path of recursion initialized as $p = [Q^T]$, the context dependent definitions of s^Q and s^{Q^*} with respect to the application context AC are defined respectively as:*

$$s^Q_{AC} = \sum_{q \in Q} f^p_{AC}(q) \tag{2}$$

$$s^{Q^*}_{AC} = \sum_{\{Q' | Q' \prec_Q Q\}} s^{Q'}_{AC} \tag{3}$$

where $f^p_{AC}(q)$ represents the relevance of the instance q of Q according to the application context AC and the recursion path p. △

Given a quality Q, an instance q of Q, and the path of recursion p starting in Q^T, $f^p_{AC}(q)$, which is the relevance of q with respect to the application context AC and p, is defined recursively with respect to the relevance of the instances of the classes reachable through p, and is formalized by the following definition.

Definition 4 (Relevance of an instance in a recursion path) *Given an application context AC, an instance ι, a path of recursion p,*

$$f^p_{AC}(\iota) = \frac{\sum_{\{a \in AC_A(p)\}} f^{p,a}_{AC}(\iota) + \sum_{\{r \in AC_R(p)\}} f^{p,r}_{AC}(\iota)}{|AC_A(p)| + |AC_R(p)|} \tag{4}$$

where $f^{p,a}_{AC}(\iota)$ and $f^{p,r}_{AC}(\iota)$ are, respectively, the relevance of an attribute a and a relation r of ι. △

The relevance of instances' properties, i.e., attributes and relations, defined for an instance in a path of recursion with respect to an application context, is formalized by the following definition.

Definition 5 (Relevance of instances' properties) *Given a path of recursion p, an instance ι of a class reachable through p, let:*

- *$x \in A \cup R$, where A is the set of attributes and R is the set of relations of ι;*
- *X a placeholder that works as a metasymbol in the formula $f^{p,x}_{AC}(\iota)$ that can be replaced by R or A if x is respectively a relation or an attribute;*
- *$i_A(\iota, a) = \{v \in V \mid (\iota, v) \in l_A(a), \exists y \in C \ s.t. \ \sigma_A(a) = (y, T) \wedge l_T(T) = 2^V\}$ the set of values assumed by the instance ι for attribute a;*
- *$i_R(\iota, r) = \{\iota' \in l_c(c') \mid \exists c \ \iota \in l_c(c) \ \exists \ c' \ s.t. \ \sigma_R(r) \in (c, c') \wedge (\iota, \iota') \in l_R(r)\}$ the set of instances related to the instance ι by relation r;*
- *AC the application context defined according to the restrictions defined in Section 3;*
- *g a function provided by the data layer, w the function parameters that have been already fixed in the application contexts:*

$$f^{p,x}_{AC}(\iota) = \begin{cases} g(v, w) & if \ (x, g(?, w)) \in AC_A(p), v \in i_A(\iota, x) \\ |i_X(\iota, x)| & if \ (x, Count) \in AC_X(p) \\ \sum_{\bar{\iota} \in i_R(\iota, x)} f^{\bar{p}}_{AC}(\bar{\iota}) & if \ (x, WCount) \in AC_R(p), \bar{p} = p \circ x \\ \frac{1}{|i_X(\iota, x)|} & if \ (x, InvCount) \in AC_X(p) \wedge i_X(\iota, x) \neq \emptyset \\ 1 & if \ (x, InvCount) \in AC_X(p) \wedge i_X(\iota, x) = \emptyset \end{cases} \tag{5}$$

△

Let's consider an application scenario where a designer navigates a repository of product models, organized according to the ontology schema of Figure 1, searching for suggestion for new products. In the following examples, we discuss a set of application contexts that can be employed to parameterize the extraction of semantic granularities in order to demonstrate the expressiveness and versatility of the proposed formalism. Examples 2 and 3 in particular provide an idea of how to apply the formulas described in this section considering different contexts.

Example 2. The user wants to navigate the product models represented in ontology the schema in Figure 1 and according to the tasks their are designed for illustrated in Figure 2. Thus, the class `Object` corresponds to S, and the hierarchy of qualities corresponds

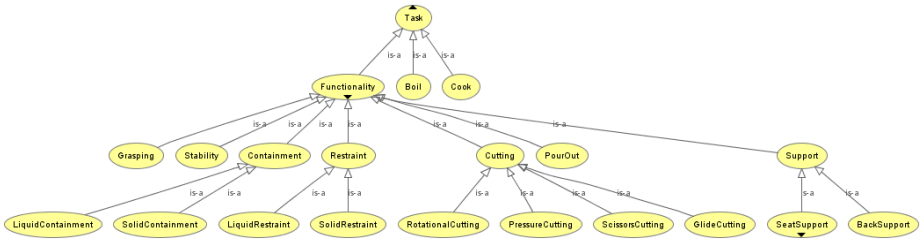

Fig. 2. The Task taxonomy

to the hierarchy having the class `Task` as Q^T. Supposing the ontology driven method corresponding to the semantic granularity is the i^{th} of the framework, the context AC_i^1 to organize the `Object` instances according to the more relevant `Task` is:

$$[Task] \overset{AC_i^1}{\rightarrow} (\{\}, \{(hasTask^{-1}, Count)\})$$

For each class $Q \in \mathcal{Q}$, its abstraction capability R_Q (see Definition 2) is evaluated according to the context AC_i^1. Applying Definition 3, s^Q and s^{Q*} are worked out initializing $p = [Task]$ as starting recursion path. The Definition 4 is recalled as $f_{AC_i^1}^{Task}$ considering as ι each instance q of Q. Thus, due to the image of the context function AC_i^1, i.e., $(hasTask^{-1}, Count)$, the application of Definition 5 is $f_{AC_i^1}^{[Task],hasTask^{-1}}(q)$, considering the relation $hasTask^{-1}$ as x and $Count$ as operation, then the second alternative in the formula $|i_R(q, hasTask^{-1})|$ is applied, counting the instances of class `Object` that are connected by $hasTask^{-1}$ to q. □

Example 3. Let's suppose the user wants to navigate the objects in the repository considering their implemented task, and taking more into account the objects with a limited weight. The context for the definition of the corresponding granularity is as follows:

$$[Task] \overset{AC_i^2}{\rightarrow} (\{\}, \{(hasTask^{-1}, WCount)\})$$
$$[Task, hasTask^{-1}] \overset{AC_i^2}{\rightarrow} (\{(weightkilos, g(?, w))\}, \{\}).$$

Where $g(?, w)$ is a function provided by the data layer, which gives a score as higher as the weight of object is less than a specified w. In this case, once Definitions 2, 3, and 4 are applied, the context AC_i^2 will force to consider the third alternative evaluating $f_{AC_i^2}^{[Task],hasTask^{-1}}(q)$. Thus, we obtain $\sum_{\bar{\iota} \in I_R(q,hasTask^{-1})} f_{AC_i^2}^{\bar{p}}(\bar{\iota})$ with $\bar{p} = [Task, hasTask^{-1}]$ and $x = hasTask^{-1}$, which brings to summing the recursive evaluation of $f_{AC_i^2}^{\bar{p}}(\iota'')$ for each instance ι'' of class `Task` connected to the initial q. Finally, for each ι'', the formula $g(?, w)$ is applied to the values of the attribute `weightKilos` bringing to consider as more relevant the tasks whose associated objects are lighter. □

Example 4. Then, suppose the user wants to navigate the objects described through the ontology in Figure 1 with respect to their implemented tasks considering more relevant the tasks associated to objects composed by different functional parts. The application context for the definition of the semantic granularity is defined as follows:

$$[Task] \rightarrow (\{\}, \{(hasTask^{-1}, WCount)\})$$
$$[Task, hasTask^{-1}] \rightarrow (\{\}, \{(hasPart, Count)\}).$$

Let suppose the user wants to navigate the objects according to their implemented tasks considering more relevant the tasks associated to light objects composed by few functional parts. The application context to be adopted in the granularity extraction is:

$$[Task] \rightarrow (\{\}, \{(hasTask^{-1}, WCount)\})$$
$$[Task, hasTask^{-1}] \rightarrow (\{(weightkilos, g(?, w))\}, \{(hasPart, WCount)\})$$
$$[Task, hasTask^{-1}, hasPart] \rightarrow (\{\}, \{(hasSubPart, InvCount)\}).$$ □

5 Conclusions and Future Works

In this paper we have proposed a formalization of contexts suitable to parameterize methods for the exploitation of knowledge encoded in ontologies, enhancing the ontology driven methods flexibility. Different examples for industrial product design repositories are provided to show how granularity needs to be explicitly parameterized with respect to the context. The examples give a sight of how to applies the formulas of the granularity and show the expressiveness of the formalization defined. The proposed approach is similar to the formalization proposed in [6,10], since our notion of context requires the specification of the features of interest for the method parameterization. However, differently from [6,10], our context specification involves specific datatype functions and context layer operations to be applied to instances' properties. Moreover, differently from [4,5,7,8,2,9], we employ contexts to parameterize ontology-driven methods that exploit background knowledge according to the specific user information needs. So far, the context as explicitly parameterization of ontology driven methods has been demonstrated as essential for the semantic similarity and the granularity. Further ontology driven methods and experimentations will be considered in the future to improve this validation.

Acknowledgement

This work is partially supported by the AIM@SHAPE Network of Excellence funded by the European Commission under the Contract IST 506766. The work of Elena Camossi is supported by the Irish Research Council for Science, Engineering and Technology

References

1. M.A. Sicilia: Metadata, semantics, and ontology: providing meaning to information resources. In *Int'l Journal of Metadata, Semantics and Ontologies*, 2006.
2. P. Bouquet, F. Giunchiglia, F. Van Harmelen, L. Serafini, H. Stuckenschmidt: Contextualizing Ontologies. In *Journal of Web Semantics*, 1:325-343, 2004.

3. M. Bazire, P. Brézillon: Understanding Context Before Using It. In Proc. of the *5th Int'l and Interdisciplinary Conf. on Modeling and Using Context*, LNAI, 3554:29-40, 2005.
4. N. Hernandez, J. Mothe, C. Chrisment, D. Egret: Modeling context through domain ontologies. In *Information Retrieval*, 10:143-172, 2007.
5. D. Vallet, P. Castellis, M. Feérnandez, P. Mylonas, Y. Avrithis: Personalized Content Retrieval in Context Using Ontological Knowledge. In *IEEE Transaction on Circuits and Systems for Video Technology*, 17(3):143-172, 2007.
6. R. Rajagopalapillai, E. Chang, T.S. Dillon: Ontology Views: A Theoretical Perspective. In Proc of the *2nd Int'l IFIP Workshop On Semantic Web & Web Semantics*, LNCS, 4278:1814-1824, 2006.
7. A. Schmidt: Ontology-Based User Context Management: The Challenges of Imperfection and Time-Dependence. In Proc. of the 5*th* Int'l Conf. on Ontologies, DataBases, and Applications of Semantics, LNCS, 4275:995-1011, 2006.
8. T. Levashova, M. Lundqvist, M. Pashkin: Moving Towards Automatic Generation of Information Demand Contexts: An Approach Based on Enterprise Models and Ontology Slicing. In Proc. of the *5th Int'l Conf. on Ontologies, DataBases, and Applications of Semantics*, LNCS, 4275:1012-1019, 2006.
9. A. Segev, A. Gal: Putting Things in Context: A Topological Approach to Mapping Contexts and Ontologies. In Proc. of the *20th National Conf. on Artificial Intelligence, 1st Int'l Workshop on Context and Ontologies: Theory, Practice and Applications*, 2005.
10. N.F. Noy, M.A. Musen: Specifying Ontology Views by Traversal. In Proc. of the *3rd Int'l Semantic Web Conference*, LNCS, 3298:713-725, 2004.
11. R. Volz, D. Oberle, R. Studer: Implementing views for light-weight web ontologies. In Proc. of *IEEE Database Engineering and Application Symposium*, IEEE Computer Society, 2003.
12. A. Magkanaraki, V. Tannen, V. Christophides, D. Plexousakis: Viewing the semantic web through RVL lenses. In Proc. of the *2nd International Semantic Web Conference*, LNCS, 2870:96-112, 2003.
13. R. Albertoni, M. De Martino: Semantic Similarity of Ontology Instances Tailored on the Application Context. In Proc. of the *5th Int'l Conf. on Ontologies, DataBases, and Applications of Semantics*, LNCS, 4275:1020–1038, 2006.
14. R. Albertoni, E. Camossi, M. De Martino, F. Giannini, M. Monti: Semantic Granularity for the Semantic Web. In Proc. of the *2nd Int'l IFIP Workshop On Semantic Web & Web Semantics*, 4278:1863-1872, 2006.
15. E. Camossi, M. Bertolotto, E. Bertino: A Multigranular Object-oriented Framework Supporting Spatio-temporal Granularity Conversions. In *International Journal of Geographical Information Science*, 20(5):511-534, 2006.
16. F.T. Fonseca, M.J. Egenhofer, C.A. Davis, G. Camara: Semantic Granularity in Ontology-Driven Geographic Information Systems. In *Annals of Mathematics and Artificial Intelligence, Special Issue on Spatial and Temporal Granularity*, 36(1-2):121-151, 2002.
17. H.R. Schmidtke: Granularity as a Parameter of Context. In Proc. of the *5th International and Interdisciplinary Conference on Modeling and Using Context*, LNAI, 3554:450-463, 2005.
18. M. Ehrig, P. Haase, M. Hefke, N. Stojanovic: Similarity for Ontologies - A Comprehensive Framework. In Proc. of the *13th European Conf. on Information Systems, Information Systems in a Rapidly Changing Economy*, 2005.