Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 475–482
ISBN 978-83-60810-51-4

# Towards Context-Semantic Knowledge Bases

Krzysztof Goczyla, Aleksander Waloszek, Wojciech Waloszek
Faculty of Electornics, Telecomunications and Informatics, Gdansk University of Technology
Email: {kris, alwal, wowal}@eti.pg.gda.pl

***Abstract*—Within the paper we discuss the issue of designing well-founded contextual knowledge bases. Following the base idea that contextualization is a vital part of conceptualization, we extend the definitions of selected notions of OntoClean, the well-known method of assessment of taxonomies, towards contextual approach. This allows us to formulate a set of desirable qualities for *context-semantic knowledge bases*. In the further part of the paper we show that SIM method, our proposal of organizing contextual knowledge base, is in accordance with the introduced desiderata.**

## I. Introduction

MODULARIZATION of knowledge bases is perceived as a modern way of defining the proper knowledge base architecture. Splitting knowledge into fragments should solve many problems faced by knowledge engineers during their works with ontologies. Modularization techniques should make reasoning more effective and reusing easier. The majority of these techniques are based on the notion of context.

Until today many modularization techniques were proposed. Almost all of them try to solve the problems with contextual reasoning, how should interpretations of modules affect each other, how to preserve decidability, soundness and completeness. Although very important, these works do not help with the problem which we call the problem of *contextual designing*. They have no tools allowing to express why contexts were created and what is their purpose. In our opinion an efficient and appropriate framework for designing *context-semantic knowledge bases*, i.e. knowledge bases where contexts are explicit elements of conceptualization and affect the semantics of other members of the model, is desirable.

In this paper we try to justify this opinion. In Section 2 we describe shortly the state of the art dividing it into four subsections. The first subsection characterizes the approach concentrated on problems of contextual reasoning, the next subsections present considerations important in our opinion for the approach connected with semantics of contexts: the metaphor of a context as a box and the method of evaluation of ontologies OntoClean. In Section 3 we present the proposition of our method of building context-semantic knowledge bases and show its semantic features describing how it corresponds to the metaphor of context as a box and how to adapt to it the rules taken from the OntoClean. Section 4 presents the summary of related works. Section 5 concludes the paper.

This work is partially supported by the Polish National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: „Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

## II. Contextual Ontologies

In this section we analyze two approaches to the problem of contextualization: the first focused on the problem of contextual reasoning and the second connected with the problem of semantic contextual designing.

### A. Principles of Contextual Reasoning

The most known modularization technique (or rather a family of techniques), DDL [1], is an adaptation of the context deductive system called DFOL (Distributed First Order Knowledge) [2] in Description Logics. This family of techniques were defined taking into consideration two *principles of contextual reasoning* given in [3]. The first principle, called the *first principle of context reasoning* (PCR1) or shortly the *principle of locality*, says that reasoning may always be performed from the perspective of a single module only and all logical consequences of this reasoning must not be inconsistent with any axiom contained in this module. The second principle, called the *second principle of context reasoning* (PCR2) or the *principle of compatibility*, says that all the logical consequences of reasoning from the given module should conform *structural constraints*, which are restrictions imposed upon the module by another modules defined in the distributed system. In DDL (like in DFOL) the structural constraints are called *bridge rules*. Describing semantic dependencies between concepts and roles defined in a pair of modules, the bridge rules say nothing about semantic of the modules (contexts) themselves. They are nothing more than low level instructions for reasoning algorithms and they do not explain the reason why contexts are created.

The idea behind such regulations is that the authors consider two types of contexts: *pragmatic* and *cognitive* [4]. The former type is defined as a point in an actual (being a part of the real world) multidimensional space where sentences are produced, and where the content, meaning and logical value of these sentences is determined. The latter one is defined as a part of the cognitive state of an agent. It is “a set of implicit assumptions that affect cognitive processes, like communication, problem solving, common sense reasoning, and so on” [4]. The consequence of such an approach is that all the related methods are suitable to solve the problem of merging of heterogeneous knowledge (or data-) bases and useful in multi-agent systems, where every agent has its own ontology and constitutes its own module (one agent—one module). It also seems to be very useful when the purpose is to reuse existing ontologies in a new project.

### *B. Need for Contextual Designing*

But it seems to be obvious that also normal, single ontologies should have modular structure. The need of constructing such ontologies follows from the observation that any information, if complex, should be ordered in a structure making designing to be more controllable. It was clearly stated by M. Minsky in [5]: *It seems to me that the ingredients of most theories both in Artificial Intelligence and in Psychology have been on the whole too minute, local, and unstructured to account—either practically or phenomenologically—for the effectiveness of common-sense thought. The chunks of reasoning, language, memory, and perception ought to be larger and more structured; their factual and procedural contents must be more intimately connected in order to explain apparent power and speed of mental activities.*

Minsky’s thoughts led to the object oriented approach which showed its power in software engineering. The most advantageous in this approach is that a designer can model reality at any level of granularity hiding details inside objects. In fact, objects are structures containing another objects, and so on, until the designer stops conceptualization and decide that there should be no more details in the model. This way she/he builds a model using “chunks” of desired size and content. In the case of knowledge engineering the “chunks” could be contexts. A structure of contexts could give users and designers opportunity to emphasize some details and to hide others, to change level of abstraction or/and to take the appropriate point of view.

The principles of contextual reasoning are not sufficient in this case. As such knowledge bases are intended to be contextual from the beginning of the designing process, contextualization should be a part of a conceptualization process. It is very important task for researchers to supply designers with rules helping to make decisions how to separate contexts, what is their semantic influence, how they are mutually related.

### *C. OntoClean*

Conceptualization was discussed in details by Guarino and Welty in many publications, e.g. [6]. All of them consider the questions what does it mean: well founded ontology, when an ontology is well founded, and how to define well founded ontologies. The effect of this discussion is the method of evaluating ontologies called OntoClean. The main purpose of the OntoClean method is to supply designers with several precisely defined rules helping them to build inside an ontology a proper taxonomies of concepts. The rules suggest what criteria should be taken into consideration during deciding about inheritance relation between two given concepts. The authors enumerate four *metaproperties* of properties (concepts) affecting the decision: *identity*, *rigidity*, *unity* and *dependency*. Let us see how does it work in the case of rigidity. The definition of this metaproperty, using modal operators, is as follows:

**Definition 1 (rigidity)**
A *rigid property* is a property that is essential to *all* instances, i.e. a property $\phi$ such that: $\forall x\ (\phi(x) \rightarrow \Box\phi(x))$.
A *non-rigid property* is a property that is not essential to *some* of its instances, i.e. $\exists x\ (\phi(x) \wedge \neg\Box\phi(x))$.

An anti-rigid property is a property that is not essential to all its instances, i.e. $\forall x\ (\phi(x) \rightarrow \neg\Box\phi(x))$. □

A property of an entity is *rigid* (or *essential*) to that entity if it *must* be true of it in every possible world, i.e. if it *necessarily holds* for that entity. For example being a human is rigid, because if an entity (i.e. someone) is a human, it is a human for as long as it exists. And being a student is an anti-rigid property, because every student finally finishes his/her education.

The rule constraining designing decisions about subsumption says that an anti-rigid property cannot subsume a rigid property. Following this rule we can decide that the concept HUMAN subsumes STUDENT but never vice-versa.

As it was already stated, contextualization is a part of conceptualization. It would be valuable to consider how contextualization changes the phenomena described by OntoClean. For this purpose it is necessary to formalize the notion of context and relations between them.

### *D. Context as a Box*

Our starting point to conduct the formalization is the metaphor called “Contect as a Box”, defining contexts as a box containing sentences and described with a set of parameters. It was invented by Bar-Hillel in 1954 [7]. More detailed description was given in [3] and [8]. Following this idea we can imagine a context as it is showed in Figure 1.

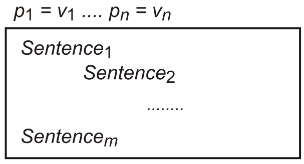


Fig. 1 Context as a box [3].

Parameters and their values replace all the complexity of the real conditions determining why a context was separated and represent all the knowledge removed outside the box. The authors of [3] describe two methods of dealing with the parameters during reasoning: *push and pop* and *shifting*. In the first method we assume that there exists a non-contextual state where the set of parameters is empty. It is an ideal state which never exists in practice. Then we can *push* some information from the inside of the box to the set of parameters. This process is called *contextualization*. The result after $n$ steps is illustrated in Figure 2 a) as the box on the left side. After the next step the information symbolically assigned $s$ is removed from all the sentences in the box and a new parameter $sit = s$ appeared in the set of parameters. The process is invertible, i.e. one can remove a parameter from the set and place an appropriate information inside the box. This process is called *decontextualization*.

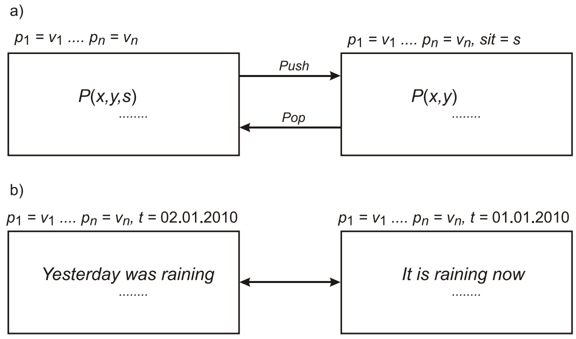


Fig. 2 Push and pop (a) and shifting (b) [3].

The second method, *shifting*, is showed in Figure 2 b). The set of parameters does not change but changing the value of a parameter causes revaluation of sentences. Like in the picture, changing the date from $1^{st}$ to $2^{nd}$ January makes the sentence “It is raining now” false.

By comparing two sets of parameters belonging to a pair of contexts and their values, we can determine a relationship the both contexts are in. For example if we have the contexts $c_1$ and $c_2$ such that their sets of parameters are equal and the values in the case of $c_2$ are either equal or more specific than in the case of $c_1$ (shifting), then we can say that $c_2$ is more specialized than $c_1$. Similarly if the set of parameters of $c_1$ is a subset of the set of parameters of $c_2$ and their values are equal, then the remaining parameters for $c_2$ make this context more specialized.

Given a set of contexts, imposing some rules constraining the context parameter sets and their values it is easy to make the relation between the contexts to be a partial order. Let us call such a structure a *context ordering structure* (*COS*).The order helps to define how the contexts affect each other in the contextual knowledge bases.

Let $P = \{p_1, \ldots, p_n\}$ be a finite set of contextual parameters which are taken into consideration during designing a knowledge base. We can define the function $f_P$: $\mathcal{K} \longrightarrow \mathcal{P}(P)$ assigning subset of $P$ to every context $k \in \mathcal{K}$. Assigning values to the parameters from the finite set $P = \{p_1, \ldots, p_n\}$ has the same meaning as building a *valuating tuple* $\sigma = \{(p_1, v_1), \ldots, (p_n, v_n)\}$, every value $v_i$ ($i \in [1..n]$) taken from the domain $\Delta_{pi}$ containing the values admissible for the parameter $p_i$. $\sigma(p_i)$ means the value assigned to the parameter $p_i$ in the valuating tuple $\sigma$. All the admissible valuating tuples make the relation $\Sigma \subset \Pi_{p \in P} \Delta_p$.

Now, if we want to speak about relationships between contexts, we have to impose a partial order on every domain $\Delta_{pi}$. Let assume that the order inside the domain $\Delta_{pi}$ is realized by the compatibility relation $\Rightarrow_{pi}$ with the maximal element $max(\Delta_{pi})$ such that for every $v \in \Delta_{pi}$ $v$ *is compatible* with $max(\Delta_{pi})$ (denoted $v \Rightarrow_{pi} max(\Delta_{pi})$). We say that $\sigma_1$ is compatible with $\sigma_2$ (denoted $\sigma_1 \Rightarrow \sigma_2$) iff $\forall p \in P$ $\sigma_1(p) \Rightarrow_p \sigma_2(p)$.

This order is intuitive and gives us the opportunity to establish a space where every point represents one context.

### *E. Rigidity in contextualized conceptualization*

Now let us consider how the rules proposed in OntoClean should be modified if applied in the space of contexts. Guarino and Welty say that a property $\phi$ is rigid iff $\forall x\ (\phi(x) \rightarrow \Box\phi(x))$. They use quantified modal logic to express that in case of rigid property it is true for an entity $x$ in every possible world. It is correct, due to the assumption of rigid designator saying that the individual name is identically interpreted in every possible model. Loosing rigidity is always connected with updating information contained by an ontology due to the changes observed in the domain being described. So if a designer assumes that a property $\phi$ is rigid, he/she expects that no future update will remove a sentence $\phi(x)$ if such a sentence once were added. And, inversely, if a property is non-rigid he/she expects that such a change will occur in the future. A new state of the world, bringing the need of updates, creates a new context which, according to the metaphor of “a context as a box”, can be described with a set of parameters. In the case of flat (not modularized) ontologies there is no habit to do it, but it is obvious that subsequent versions of a given ontology can be described along the dimensions characteristic for contexts (e.g. partiality, approximation, perspective). Therefore there exists such a set of parameters $P = \{p_1, \ldots, p_n\}$, which is able to contain information sufficient to describe every future version of this ontology. Every expected change inserted into the ontology reflects a given possible world and may be represented by the relation Σ, described above. So if we assign our ontology with the letter $O$ it is convenient to denote the version described by the tuple $\sigma \in \Sigma$ by $O(\sigma)$. Let $\mathcal{S}(O)$ means the signature of $O$. Now we can define rigidity as follows:

**Definition 2 (rigidity in ontology)**
A property ϕ is rigid in ontology $O$ iff for every pair of its versions $O(\sigma)$, $O(\sigma')$ and any $x$ belonging to the signatures of the both versions, $\phi(x)$ is entailed either by the both versions or by none. Formally:

$$\forall x\ (\forall \sigma,\sigma' \in \Sigma\ (x \in \mathcal{S}(O(\sigma)) \cap \mathcal{S}(O(\sigma')) \rightarrow (O(\sigma) \vDash \phi(x) \leftrightarrow O(\sigma') \vDash \phi(x))))$$ □

Describing expected versions with a set of parameters allows us to ask how chosen subsets of parameters affect rigidity. Some properties may be rigid according to one subset of parameters and non-rigid according to others. For instance the temperature of evaporation of water does not change in time but depends on pressure. Let us call this kind of rigidity *local rigidity*. Let $\sigma[P']$, where $P' \subseteq P$, means a subtuple containing valuation of parameters from $P'$ only: $\sigma[P'] = \{(p_i, v_i): (p_i, v_i) \in \sigma \wedge p_i \in P'\}$.

**Definition 3 (local rigidity w.r.t. a subset of parameters)**
A property ϕ is locally rigid in ontology $O$ w.r.t. the subset of parameters $P' \subseteq P$ iff for every pair of its versions $O(\sigma)$, $O(\sigma')$ such that $\sigma[P'] = \sigma'[P']$, and any $x$ belonging to the signatures of the both versions, $\phi(x)$ is entailed either by the both versions or by none. Formally:

$$\forall x\ (\forall \sigma,\sigma' \in \Sigma\ ((x \in \mathcal{S}(O(\sigma)) \cap \mathcal{S}(O(\sigma')) \wedge \sigma[P'] = \sigma'[P']) \rightarrow (O(\sigma) \vDash \phi(x) \leftrightarrow O(\sigma') \vDash \phi(x))))$$ □

We can say that local rigidity is rigidity in a subset of the set of possible worlds. All possible worlds from this subset are described with the parameters substituted in such a way that parameters from the subset $P'$ have fixed values. The *global rigidity* is a special case of the local one—it is a local rigidity for $P' = \varnothing$.

There is no need to analyze the remaining metaproperties as their contextual properties are similar. In the same way we can describe *local identity*, *local unity* and *local dependency*.

Different kinds of updates "move" an ontology through the space of Σ, changing some values of parameters and preserving the others. The substantial question here is what does it mean "a new version of $O$", or what kind of changes just "maintain" an ontology giving its new versions, but not new ontologies. An intuitive answer is that the only changes maintaining ontologies are monotonic changes. But in the case of flat ontologies it is very difficult to fulfil this requirement. Thus differentiation between a new ontology and a new version of an ontology is very vague.

Contextual ontologies are more capable to retain monotonicity because they are able to cope with contradictions. But they do it because they contain more than one context. We can say that while a flat ontology is connected always with one point of Σ, a contextual ontology engages a subset $\Sigma_i \subset \Sigma$. Inside $\Sigma_i$ it is rationally to speak about locality of the metaproperties.

Taking it into consideration we can now define what semantics of contexts means. Semantics of contexts is a function assigning to every pair $(k, \phi)$, where k is a context and ϕ is a property, values of every metaproperty defined by OntoClean. So, in other words, we can say that contextual designing determines semantics of contexts by defining the range of locality of the metaproperties for every defined property.

We can also say that *context-semantic knowledge base* (CSKB), is a knowledge base which is able to preserve the definition of the range of locality of the metaproperties during its lifetime.

## III. SIM Method

In this section we present our proposition of a context-semantic knowledge base. We introduced it in [9] where it was called a method of hierarchical contextualization of ontologies. Later on the method was called Structural Interpretation Model. After a short presentation we will show how it fulfils the definition of CSKB.

### A. Description of the Method

The structure of a contextual ontology designed according to the SIM rules consists of two parts: a terminology part, called contextualized TBox (or contextualized terminology) and a factual part called contextualized ABox (or contextual description of the world). Symbolic names TBox and ABox are taken from Description Logics but the kind of logic is not obligatory except it must be able to distinguish general statements (quantified) and detailed statements (description of facts). The formal definition is as follows:

**Definition 4 (contextual TBox)**
A *contextualized TBox* (shortly *c-TBox*) is a structure $\mathbf{T} = (\mathcal{T}, \trianglelefteq, T_{max})$ *and* consists of a set of TBoxes $\mathcal{T}$ whose elements are called *context types*, and a *generalization relation* $\trianglelefteq \subseteq \mathcal{T} \times \mathcal{T}$ which is a partial order established over the set of $\mathcal{T}$. The structure $\mathbf{T}$ contains the maximal element $T_{max} \in \mathcal{T}$, such that $T \trianglelefteq T_{max}$ for every $T \in \mathcal{T}$.
We also introduce the following notions ($T_1, T_2, T_3 \in \mathcal{T}$):

- $T_1$ *generalizes/is an ancestor of* $T_2$ iff $T_2 \trianglelefteq T_1$,
- $T_1$ *specializes/is a successor of* $T_2$ iff $T_1 \trianglelefteq T_2$.

- A *direct generalization* $\trianglelefteq\bullet$ is a subset of $\trianglelefteq$ such that $T_1 \trianglelefteq\bullet T_2$ iff $T_1 \trianglelefteq T_2 \wedge T_1 \neq T_2 \wedge \neg\exists T_3\ (T_1 \trianglelefteq T_3 \wedge T_3 \trianglelefteq T_2)$.

As a TBox $T \in \mathcal{T}$ contains a set of sentences, the generalization relation describes how knowledge is distributed between the elements. More specialized terminology imports more general one, but not vice-versa. It works in the same manner as owl:import predicate in OWL. The symbols $\mathcal{S}_p(T)$ and $\mathcal{S}_t(T)$ represent respectively the *signature of predicates* and *signature of terms* of *T* after the import, i.e. if $T_1 \trianglelefteq T_2$, then $\mathcal{S}_p(T_2) \subset \mathcal{S}_p(T_1)$ and $\mathcal{S}_t(T_2) \subset \mathcal{S}_t(T_1)$. □

It is worth noting that there is no flow of information between context types not related by the generalization relation. Therefore, if in such contexts exist the same predicate names not inherited from a common ancestor, they have not the same meaning unless they have a common successor. If they have one, then the same predicate names have the same meaning but only in the domain of the successor (see Definition 7).

**Definition 5 (contextual ABox)**

A *contextualized ABox* (shortly *c-ABox*) $\mathbf{A} = (\mathcal{A}, \lessdot, inst, A_{max})$ of contextualized TBox $\mathbf{T} = (\mathcal{T}, \trianglelefteq, T_{max})$ is a structure consisting of:

1) A set of ABoxes $\mathcal{A}$, each of which is called an *instance of context*. As an ABox $A \in \mathcal{A}$ contains a set of sentences, the symbols $\mathcal{S}_p(A)$ and $\mathcal{S}_t(A)$ represent respectively its *signature of predicates* and *signature of terms*.
2) The *aggregation relation* $\lessdot \subseteq \mathcal{A} \times \mathcal{A}$, which is a partial order established over the set of $\mathcal{A}$.
3) The maximal element $A_{max} \in \mathcal{A}$, such that $A \trianglelefteq A_{max}$ for every $A \in \mathcal{A}$.
4) The function *inst*: $\mathcal{A} \rightarrow \mathcal{T}$ relating every instance *A* from $\mathcal{A}$ with a context type *T* from $\mathcal{T}$. We require that:
   a) $\mathcal{S}_p(A) \subset \mathcal{S}_p(inst(A))$
   b) $inst(A_{max}) = T_{max}$,
   c) for each $A_1 \lessdot A_2$ such that $A_1 \neq A_2$ holds $inst(A_1) \trianglelefteq inst(A_2)$ and $inst(A_1) \neq inst(A_2)$.

We also say that $(A, A_1, A_2, \in \mathcal{A})$:

- *A* is an *instance* of the context type *T* iff $inst(A) = T$,
- $A_1$ *aggregates* $A_2$ iff $A_2 \lessdot A_1$,
- $A_1$ *is aggregated by* $A_2$ iff $A_1 \lessdot A_2$,
- $A_1$ is an *aggregating context instance* iff $\exists A_2$: $A_2 \lessdot A_1$.

□

The both parts of a contextual ontology, c-TBox and c-ABox, have their own substructures built by the $\trianglelefteq$ and $\lessdot$ relations. The *inst* function links them together in such a way that several ABoxes may be assigned to a single TBox. (More detailed explanation of the assumptions described in Def. 5 can be found in [9].) Figure 3 shows an example of a contextual knowledge base designed according to the SIM method. The structure of this base contains three context instances: $A_7$, $A_8$, and $A_9$ which are assigned to a single context type (namely $T_5$). Relationships between context instances and between context instances and contexts are depicted in the form of a graph, e.g. the instance $A_4$ aggregates instances $A_8$ and $A_9$. For the sake of clarity only the transitive reductions of generalization and aggregation relations have been depicted.

It gives us distinctive opportunities: different ABoxes may contain different (consistent locally but possibly inconsistent with other ABoxes) sets of assertions describing different groups of individuals using the same notions (concepts and roles). A pair $\kappa = (T, A)$ such that $inst(A) = T$ is called a *context*. By $T(\kappa)$ and $A(\kappa)$ we denote the first and the second element of the pair respectively.

The both relations ($\trianglelefteq$ and $\lessdot$) and the function *inst* determine the paths information flow along. In the case of aggregation relation the flow has the direction from aggregated to aggregating instances. All sentences must be taken into account, but due to the fact that the attached TBox is more general some information must be reinterpreted in more general terms. Figure 3 depicts three aggregating context instances: $A_1$, $A_3$ and $A_4$.

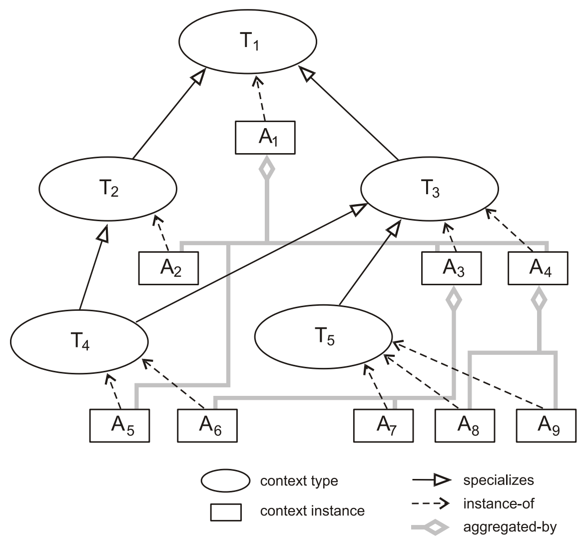


Fig. 3 An example of a SIM knowledge base.

If levels of generality (introduced by context types) are properly chosen, we can aggregate information from context instances holding contradictory assertions without making the knowledge base inconsistent.

The main principle of SIM structures is that the higher level of hierarchy the larger set of individuals is described in the more general way.

The maximal element of aggregation relation aggregates all the other context instances. It means that all the context instances of a given c-ABox have to be consistent with each other at the highest level of generality. Thus this element consolidates all the information hidden in all the modules. The both structures, c-TBox and c-ABox, together make a contextualized knowledge base:

**Definition 6 (contextual knowledge base)**
A *contextualized knowledge base* **K** = (**T**, **A**) consists of a c-TBox **T** and a c-ABox **A** of **T**. □

Sometimes it is convenient to perceive **T** as a *schema* of a knowledge base and A as its *instance*. Thus the set of all possible knowledge bases $\mathcal{R}_\mathbf{K}$ is a relation $\mathcal{R}_\mathbf{K} \subset \mathcal{R}_\mathbf{T} \times \mathcal{R}_\mathbf{A}$, where $\mathcal{R}_\mathbf{T}$ is the set of all possible schemas and $\mathcal{R}_\mathbf{A}$ is the set of all possible instances. Obviously not every $\mathbf{A} \in \mathcal{R}_\mathbf{A}$ may be an instance of a given $\mathbf{T} \in \mathcal{R}_\mathbf{T}$. A pair (**T**, **A**) may be an element of $\mathcal{R}_\mathbf{K}$ iff it fulfils Definition 4. The set of all admissible instances of **T** is denoted by $\mathcal{A}_\mathbf{T}$.

The definition of the ⊴ and ≪ relations and the *inst* function determines the direction of information flow. The next definition explains how the flow should be performed to fulfil the principles of locality and compatibility.

**Definition 7 (interpretation and satisfiability)**
A *contextualized interpretation* (shortly c-interpretation) of contextualized knowledge base **K** = (**T**, **A**), where **T** = ($\mathcal{T}$, ⊴, $T_{max}$) and **A** = ($\mathcal{A}$, ≪, *inst*, $A_{max}$), is a set of interpretations $\mathcal{J}$.
We say that $\mathcal{J}$ satisfies **K** ($\mathcal{J} \vDash \mathbf{K}$) iff:
1) For every individual name *a*, there do not exist two interpretations $\mathcal{I}_1, \mathcal{I}_2 \in \mathcal{J}$ such that $a^{\mathcal{I}_1} \neq a^{\mathcal{I}_2}$.
2) For every context instance $A_1 \in \mathcal{A}$:
   a) $\exists \mathcal{I} \in \mathcal{J}$ such that $\mathcal{I} \vDash A_1$, denoted by $\mathcal{J}[A_1]$,
   b) $\mathcal{J}[A_1] \vDash \bigcup\{T \in \mathcal{T}: inst(A_1) \trianglelefteq T\}$,
   c) for every $A_2$ such that $A_2 \ll A_1$:
      i) $\Delta^{\mathcal{J}[A_2]} \subseteq \Delta^{\mathcal{J}[A_1]}$,
      ii) for every predicate $C$ of arity $n$ from $\bigcup\{T \in \mathcal{T}: inst(A_1) \trianglelefteq T\}$: $C^{\mathcal{J}[A_2]} = C^{\mathcal{J}[A_1]} \cap (\Delta^{\mathcal{J}[A_2]})^n$. □

The convention of symbols used in this definition, corresponds to the widely used standard definition of first order interpretation: $\mathcal{I} = (\Delta^\mathcal{I}, \bullet^\mathcal{I})$: $a^\mathcal{I}$ means an element of $\Delta^\mathcal{I}$ assigned to the individual name *a* and $C^\mathcal{I}$ means a subset of $(\Delta^\mathcal{I})^n$ assigned to the predicate name *C* (of arity *n*) by the function $\bullet^\mathcal{I}$.

The rule 1 introduces *uniformity of names*. It ensures that a single individual name used throughout the knowledge base is always locally interpreted in the same way.

The rules 2.a and 2.b, let us call them *local conformance rules*, ensure preservation of the principle of locality. Each local interpretation satisfies the ABox and the TBox of the context instance it is assigned to and all TBoxes being its ancestors.

Other rules follow the principle of compatibility, i.e. introduce the desired level of interaction between interpretations. The rules 2.c establish relations between aggregating context instance and context instances being aggregated. They are called *aggregation conformance rules*. The rule 2.c.i introduces aggregation *conformance of domains* and states that the domain of the interpretation of the aggregating context must cover domains of interpretations of all context instances being aggregated. The rule 2.c.ii establishes *aggregation conformance of denotation*. It states that within the limited domain of the context instance $A_2$ being aggregated by $A_1$, at the level of generality of the terminology $T$ ($inst(A_1) = T$), all the concepts and roles must be interpreted in $A_1$ in the same way (have the same extensions) as in $A_2$.

In our further considerations we will also need the notion of entailment.

**Definition 8 (entailment)**
Entailment in contextual knowledge base **K** = (**T**, **A**) where **T** = ($\mathcal{T}$, ⊴, $T_{max}$) and **A** = ($\mathcal{A}$, ≪, *inst*, $A_{max}$):
1) ϕ is entailed by **K** in the context κ (denoted $\mathbf{K} \vDash_\kappa \phi$) iff it is entailed by κ,
2) ϕ is entailed by κ (denoted $\kappa \vDash \phi$) iff for every c-interpretation $\mathcal{J}$ that is a model of **K** it is true that $\mathcal{J}[A(\kappa)] \vDash \phi$. □

### B. SIM and Contextual Parameters

The SIM formalism does not explicitly use the notion of context parameters. But it is easy to observe that the structure built by the relations defined in the formalism reflects a structure COS. Let again $P = \{p_1, \ldots, p_n\}$be a finite set of contextual parameters which are taken into consideration during designing a knowledge base. Now we can define the function $f_P$ in a slightly different way. $f_P$: **T** ⟶ $\mathcal{P}(P)$ assigns a subset of $P$ to every context type. The main property of this function is implied by the fact that with moving down the generalization relation (walking away from the least element) the context types are increasingly specialized. Deciding only about the content of the set of parameters (without their values) the only way to make a context type more specialized is to push a new parameter into the set. Therefore the function $f_P$ should assign the parameters to the context types in such a way that:

$$\forall T_1 \in \mathbf{T}\ (T_1 \neq T_{max} \rightarrow \bigcup\{f_P(T_2): T_1 \trianglelefteq\!\bullet\, T_2\} \subseteq f_P(T_1)) \qquad (1)$$

Every specializing context type inherits all the parameters of the contexts types being specialized and, optionally, adds new ones.

Every context instance of the type $T$ has to be described by assigning values to every parameter from the set of parameters belonging to $T$. The function describing context instances is $f_V$: $\mathcal{A}_\mathbf{T} \longrightarrow \Sigma$ such that $\forall \mathbf{A} \in \mathcal{A}_\mathbf{T}\ \forall A_1, A_2 \in \mathbf{A}$, $A_1 \neq A_2$ and $f_P(inst(A_1)) = \{p_1, \ldots, p_k\}$:

$$f_V(A_1) = \{(p_1, v_1), \ldots, (p_k, v_k), (p_{k+1}, max(\Delta_{pk+1})), \ldots, (p_n, max(\Delta_{pn}))\}, \qquad (2)$$

$$f_V(A_1) \neq f_V(A_2) \text{ (the function is an injection)}, \qquad (3)$$

$$A_1 \ll A_2 \rightarrow f_V(A_1) \Rightarrow f_V(A_2). \qquad (4)$$

If the functions $f_P$ and $f_V$ are defined as above, the structure of contexts (i.e. pairs type-instance) in a SIM knowledge base is equivalent to COS. Defining the SIM relations: $\trianglelefteq$, $\ll$ and *inst*, is more convenient than assigning parameters and valuating them but brings the same effect.

### C. SIM and locality of the metaproperties

In this section our intention is to discuss how the SIM method preserves locality of the metaproperties using rigidity as an example. In the sake of simplicity we can assume that all possible versions of an ontology form a subset of $\mathcal{R}_\mathbf{K}$ such that:

- all the pairs have the same given schema **T**,
- knowledge base instances belong to $\mathcal{A}_\mathbf{T}$ and are monotonic. Let us denote them with $\mathcal{A}_\mathbf{T}^\mathrm{m}$.

As it was shown earlier, in the graph representing the generalization relation the number of exposed parameters grows when a context type is more specialized. For this reason we can bind locality of rigidity with a part of the SIM structure. Choosing a subset of parameters means in this case choosing a place in the hierarchy of context types. Choosing a parameter valuation tuple means pointing out a place in the hierarchy of context instances.

As we pointed in Section II.E a contextual knowledge base is connected with $\Sigma_i$ rather than with a single $\sigma$. The reason is that in the case of a SIM model $\sigma$ is related with a part of it, namely with such context instances $A$ for which holds $f_V(A) \Rightarrow \sigma$. It seems to be more convenient then to speak about instances. Furthermore **T** is constant. Thus we can transform the expression from Definition 3 as follows:

$$\forall x\ (\forall \mathbf{A}, \mathbf{A'} \in \mathcal{A}_\mathbf{T}\ (\forall A_1 \in \mathbf{A}\ (\forall A_2 \in \mathbf{A'}$$
$$((x \in \mathcal{S}_t(A_1) \cap \mathcal{S}_t(A_2) \wedge f_V(A_1)[P'] = f_V(A_2)[P'])$$
$$\rightarrow (A_1 \vDash \phi(x) \leftrightarrow A_2 \vDash \phi(x))))$$

We can simplify this formula noting that with the assumption of monotonicity we can take into consideration one knowledge base instance only:

$$\forall x\ (\forall \mathbf{A} \in \mathcal{A}_\mathbf{T}^\mathrm{m}\ (\forall A_1, A_2 \in \mathbf{A}$$
$$((x \in \mathcal{S}_t(A_1) \cap \mathcal{S}_t(A_2) \wedge f_V(A_1)[P'] = f_V(A_2)[P'])$$
$$\rightarrow (A_1 \vDash \phi(x) \leftrightarrow A_2 \vDash \phi(x)))))$$

It is easy to observe that the SIM structure conform this expression. The equality $f_V(A_1)[P'] = f_V(A_2)[P']$ is true for the context instances aggregated by the instances of the context type $T$ such that $P' \subseteq f_P(T)$. So we can transform the formula again:

$$\forall x\ (\forall \mathbf{A} \in \mathcal{A}_\mathbf{T}^\mathrm{m}\ (\forall A_1, A_2, A_3 \in \mathbf{A}$$
$$((x \in \mathcal{S}_t(A_1) \cap \mathcal{S}_t(A_2) \wedge A_1 \ll A_3 \wedge A_2 \ll A_3$$
$$\rightarrow (A_1 \vDash \phi(x) \leftrightarrow A_2 \vDash \phi(x))))))$$

It is obvious that according to the point 2.c.ii of Definition 7, this formula must be true for all predicates $\phi \in \mathcal{S}_p(inst(A_3))$ (not only for unary ones).

The above consideration illustrates the ability of the SIM method to preserve locality of the metaproperties. This notion, when analyzed together with contextualized knowledge bases, seems to be much more concrete, because affects concrete structural and functional features of created systems.

## IV. Related Work

The majority of works on modularization of knowledge bases do not analyze the problem of semantics of contexts. Although there are several general considerations in the metaphysical and epistemological level of abstraction (like in [4] or [3]) there are very few works trying to propose practical solutions.

To the best of our knowledge, the only solution, except the SIM method, based on the metaphor of context as a box, is the proposition of the method of building modular knowledge bases called *Contextual Knowledge Repository* [10]. CKR is a set of *contexts* **C** and the knowledge base **D** which is expressed in Description Logics. **D** defines contextual parameters, called here *dimensional attributes*, their domains and concepts denoting subsets of these domains. The set of contexts contains elements, each of which is defined as a set of sentences (expressed also in DL) about a domain of interest, and a set of roles (defined in **D** dimensional attributes) assigning *dimensional values* to the context. There are two kinds of contexts: *primitive contexts* and *context classes*. The former contain detailed information tightly connected with the point of the space which is similar to $\Sigma$. The values of parameters assigned to them are constants. The latter contain more general information and some of the parameter values assigned to them have to be concepts defined in **D**. The formulas defined inside a context class may describe the domain of interest or, using *qualified concepts*, may control the propagation of information between contexts.

A user asks CKR formulating conjunctive queries in a language allowing to specify the *dimensional vector* determining the set of contexts reviewed by the system during searching the answer. If the query contains a *total dimensional vector* then the query is *fully contextualized* and is precisely addressed to one context. If the dimensional vector is partial then the query is addressed to the set of contexts. The tuples contained in the answer are completed with the missing part of the vector.

The order imposed on the set of contexts is a step forward to organize context structure. But this structure has only weak relationship with the semantics understood as rules adjusting semantic compatibility between modules. The only way to define this compatibility are formulas with qualified concepts but they strongly resemble bridge rules from DDL and have nothing to do with contextual designing.

## III. CONCLUSION

In the paper we considered the problem of semantic designing of context-semantic knowledge bases. We tried to justify the opinion that effective, efficient and successful contextual designing is not possible without perceiving contextualization as a significant part of conceptualization. We presented the proposition of our method of building contextual knowledge bases. Then we showed how it corresponds to the metaphor of context as a box which is a foundation for formalization semantics of contexts. We also analyzed how to adapt the rules taken from the OntoClean method to contextual knowledge bases. In our opinion farther consideration of the results achieved by the authors of OntoClean in the area of contextual conceptualization may be very fruitful and may result in creating a full contextual framework for knowledge base developers.

## REFERENCES

[1] A. Borgida, and L. Serafini, "Distributed Description Logics: Directed Domain Correspondence in Federated Information Sources," in *On the Move to Meaningful Internet Systems 2002*, LNCS vol. 2519, Springer Verlag, 2002, pp. 36–53.

[2] C. Ghidini, and L. Serafini, "Distributed First Order Logics," in *Frontiers of Combining Systems 2, Studies in Logic and Computation*, D. Gabbay, and M. deRijke, Eds. Research Studies Press, 1998, pp. 121–140.

[3] M. Benerecetti, P. Bouquet, and C. Ghidini, "Contextual reasoning distilled," in *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 12, 2000, pp. 279–305.

[4] F. Giunchiglia, and P. Bouquet, "Introduction to contextual reasoning. An Artificial Intelligence perspective," in *Perspectives on Cognitive Science*, B. Kokinov, Ed., vol. 3, NBU Press, 1997, pp. 138–159.

[5] M. Minsky, "A framework for representing knowledge," in *Readings in Knowledge Representations*, R. Brachman, and H. J. Levesque, Eds., Morgan Kaufmann, 1985, pp. 245–262.

[6] N. Guarino, and C. A. Welty, "Ontological Analysis of Taxonomic Relationships," in *Proceedings of ER-2000: The 19th International Conference on Conceptual Modeling*, LNCS vol. 1920, Springer Verlag, London, 2000, pp. 210–224.

[7] Y. Bar-Hillel, "Indexical Expressions," in *Mind*, vol. 63, 1954, pp. 359–379.

[8] M. Benerecetti, P. Bouquet, and C. Ghidini, "On the dimensions of context dependence: Partiality, approximation, and perspective," in *CONTEXT '01 Proceedings of the Third International and Interdisciplinary Conference on Modeling and Using Context*, Springer Verlag, London, 2001, pp. 59–72.

[9] K. Goczyła, A. Waloszek, and W. Waloszek, "Contextualization of a DL knowledge base," in *Proceedings of the 20th International Workshop on Description DL'07*, Brixen/Bressanone, 2007, pp. 291–299.

[10] M. Homola, L.Serafini, and A. Tamilin, "Modeling Contextualized Knowledge," in *CIAO '10 Proceedings of the 2nd Workshop on Context, Information and Ontologies*.