Visualizing the Bias of Enterprise Metamodels towards Nuanced Concepts

David Naranjo, Mario Sánchez, and Jorge Villalobos
Systems and Computing Engineering Department
Universidad de los Andes
Bogotá, Colombia

***Abstract*—In Enterprise Modeling, we use several languages for designing, analyzing, and communicating the different domains of an enterprise. Two important criteria for choosing a domain-specific language are its appropriateness to the requirements of the enterprise, as well as the accuracy of the language in describing the domain at hand. However, in some business domains, core concepts –such as Capability, Service, and Value– represent constructs that have different (and often conflicting) definitions and interpretations among the literature. In this context, the bias of a language is the preference for a theory, i.e. a particular interpretation of a concept. Currently, there is no explicit way of assessing and communicating this bias, and thus it remains difficult to assess the appropriateness and accuracy of a language for a particular purpose. In this paper, we provide a method for visual comparison of this bias with regard to the Capability concept, comparing three theories and three modeling languages where this concept is pivotal.**

I. Introduction

In Enterprise Modeling, models are structured abstractions of reality that are produced with a Modeling Language. Over the years, these fields have made a transition from using general-purpose Modeling Languages (such as UML) to using more specialized languages, such as ArchiMate, BPMN, and SOAML, with the added value of having a myriad of tools and methodologies that support the modeling and analysis of models produced with these languages.

With this abundance of languages, we can model and analyze several relevant aspects of an enterprise; all we need is the disposition, time, and resources to do so. Furthermore, if these languages do not suffice, we can take fragments of them, we can extend them, or even create micro-languages that are tailored for more specific domains and purposes. Thus, the focus of attention nowadays is not in *how* to model a perspective of an enterprise, but on *which* modeling languages serve our purposes best, and how can we accommodate multiple languages to our requirements.

This has lead to a –still ongoing– discussion of what does it mean to model, and which are the desired properties of modeling languages. According to Selic [1], an engineering model must possess the following five characteristics: Abstraction, understability, accuracy, predictiveness, and inexpressiveness.

In particular, a model is *accurate* if it "provides a true-to-life representation of the modelled system's features of interest" [1]. This accuracy can be straightforward to assess for some domains of an enterprise, e.g. infrastructure, applications, and resources, where conceptual entities have a direct correspondence to objects in reality. However, when modeling business and strategic domains, this is a more difficult task: Many concepts of these domains have different interpretations, depending on the theories on which they are supported. For instance, consider the concept of **Capability**:

This construct has gained a privileged position in the discourse of strategic management, and can be colloquially explained as both the *capacity* and *ability* of doing something. First introduced in 1965 by Ansoff [2], this construct has had several comebacks in the literature: Around 1995, [3], 2005 [4], and recently, [5]. Fifty years since its inception, this concept has brought an plethora of definitions, and has led to several literature reviews (see [6] and [7]) as authors constantly revise and appropriate this concept in their theories, sometimes leading to inconsistencies between them [6]. These different theories have their own classification schemes, granularity levels, and methods of analysis. For these reasons, we consider the term Capability as a *nuanced* concept, i.e. it has different interpretations depending on the context that it is applied.

In this paper, we argue that the accuracy of a modeling language can be affected by nuanced concepts introduced to the language, as there is a potential discrepancy between the perceived semantics of a concept by language designers and its interpretation among language users, resulting in a false sense of agreement. In order to make explicit this discrepancy, we introduce the notion of *bias* of a modeling language as its tendency to use specific theories of a nuanced concept. We also propose an approach for visualizing this bias, where multiple relevant theories and languages can be examined, based on related concepts in their metamodels.

This examination is made by creating an ontology of the essential classifiers that are used to define the controversial concept, and can be used to compare languages and theories alike. This classification scheme leaves a visual signature that can be used to indicate the bias of each essential classifier.

This paper is structured as follows: Section II offers a background of the problem, centered on the relationship between Syntax and Semantics. In Section III, we formalize the notion of nuance, and provide a conceptual framework around this concept. In Section IV, we describe our approach for calculating this bias. Then, Section V demonstrates our approach for the Capability concept.

II. The Dark Side of Modeling

Being such an essential activity, *modeling* has been approached by a confluence of disciplines, using artifacts such as conceptual models, ontologies, database schemas, engineering models, and architecture models in their respective fields. However, like other central notions (such as 'information'), the term 'model' is hard to define, and there is no consistent or common understanding of what it means *'to model'* [8].

While the literature has left us with several definitions [9], it seems that "there is no apparent consensus about the terminology, not even about what should be the minimal set of basic modeling concepts." [10]. However, despite the myriad of interpretations this concept carries, we keep modeling, and conferences keep flourishing, with several research communities that use models as their main artifacts; it seems that each community adopts a particular definition, and develops theories that make use of this notion of modeling.

For this reason, the term *model* can be seen as another example of a **nuanced** concept: It has different interpretations, depending on the context and purposes that the term serves. For example, in model-based Enterprise Architecture, a Model is a description of an enterprise among several domains – such as the Strategy, Business, Application, and Technology domains– and is used for several purposes, such as analysis, design, simulation, and communication between stakeholders. These models are produced with domain-specific languages such as ArchiMate, BPMN, SOAML, i*, or Metamodels such as BMM, and the one of TOGAF.

A. Modeling Languages and Metamodels

According to Morris [11], a modeling language can be divided in three parts: Syntax, Pragmatics, and Semantics. The Syntax is described in terms of a **Metamodel**, which "makes statements about what can be expressed in the valid models of a certain modeling language" [12]. This Metamodel is usually called a *Linguistic Metamodel* [13].

Pragmatics concern the relation of signs to human interpreters [11], and can be associated to the *pragmatic feature* defined by Stachowiak [14], or the *abstraction* property of Selic: "*A model needs to be usable in place of an original with respect to some purpose*". In addition, we need that our models are reasonably inexpensive to develop, which again can be related to the *reduction feature* of Stachowiak, and the *inexpensiveness* of Selic.

Semantics are concerned with the meaning and interpretation of the concepts embedded in the language. As Harel and Rumpe [15] underscore, Semantics of a language specify the very concepts that exist in the **Universe of Discourse**, capturing decisions about the kinds of things the language should express. We can associate this aspect as the *Mapping Feature* of Stachowiak [14] ("*A model is based on an original*"), or the *Accuracy* ("*it provides a true-to-life representation of the modelled system's features of interest*") of Selic [1].

B. Syntax versus Semantics

Motivated by common misunderstandings, Harel and Rumpe [15] considered necessary to draw a clear line between the Syntax (described by a Linguistic Metamodel), and the Semantics (i.e. its interpretation) of a modeling language.

The mapping between conceptualizations of a domain and elements of a language is often found in the specifications of the language notation. Usually, concepts are described informally through examples and definitions in natural language (e.g. plain English), as there is no simple and obvious way to define this complex semantic mapping precisely, clearly, and readably [15].

As explained by Guizzardi [16], this mapping is an **Ontology** in the philosophical sense, i.e. it is a particular world-view embedded in the modelling primitives of a language. When explicit, this ontology is called an *Ontological Metamodel* [17] of the language, or simply, the *Ontology* of the language.

In regard to the relationship between Metamodels and Ontologies, Atkinson and Kühne [18] examine the roles of semantic and ontological types, and describe situations when the distinction between both types (and therefore between Syntax and Semantics) dissolves.

For instance, in EA modeling languages, the **linguistic type** *Process*, and the **ontological classifier** *Process* share an obvious lexical similarity. However, the function of the former is to produce tokens in the Language, while the latter refers to a construct, i.e. the abstract idea of a process; there is a dichotomy on the *roles* that linguistic and semantic types play.

As Atkinson and Kühne [18] explain, we can start to see these differences if we consider the context on which both types are used: " *it becomes clear that even though types in a domain-specific language definition could have the appearance of ontological types, they do not in any way fulfill the same function*".

Nevertheless, the authors point out that the intended use of a type may not be entirely clear, notwithstanding any semantic litmus test that is performed. More importantly, they suggest that such ambiguity could be the basis for an approach that promotes **ambivalent classification** as a feature, and "believe that future work should provide a comprehensive analysis of the trade-offs involved in using [this] approach".

Thus, the choosing of linguistic classifiers in a notation is not arbitrary, and holds a relationship to ontological classifiers that represent the conceptualization and interpretation of a domain. In this manner, the relationship between Syntax and Semantics is the 'dark side of modelling', not because it is ill-defined, but because it is usually *hidden*.

In the context of Semantic Interoperability, Karagiannis and Höfferer [19] introduce the term *Inherent Semantics*, which are "a kind of *inner meaning* of modeled resources that is exceeding the type semantics that is being inherited by the elements of the metamodel-layer". On a related manner, but in a different field, Third [20] defines the *hidden semantics* of an ontology as the given names or labels of ontology identifiers, "corresponding to natural language words or phrases which are very dense with information as to their intended referents".

C. Semantic Structures and the Ontology Spectrum

Nowadays, it is acknowledged that modeling languages and ontologies complement each other and both are needed to describe knowledge for a complex domain [17], [19], [21]–[23]. Furthermore, modeling languages such as the Semantics of Business Vocabulary and Rules (SBVR) language [24], try to mix both worlds, providing a model of the meanings and interpretations of concepts for a specific organization.

It is common to refer to an ontological model as *the ontology* of a domain. However, given the liberal use of this term in the literature, researchers have coined the notion of a ontological *spectrum* [22] or continuum [25] to describe a range of different semantic artifacts (e.g. controlled lists of words, taxonomies, thesauri, topic maps) that are commonly used to describe concepts and their relationships.

The difference, as with models, mostly lies in their purpose: (Formal) ontologies aim to describe categories of existence [26], and to organize canonical knowledge [25] with clearly defined hierarchical relationships and constraints, in order to reason and make inferences.

Conversely, other semantic structures, such as terminologies, are structured vocabularies, i.e. organized sets of terms that reflect the "most common" definitions of a concept, and are used for unambiguous communication in natural language.

In particular, **thesauri** are useful to disambiguate meanings of similar concepts, and usually offer a longer set of relationships (e.g. meronymy, hyponymy, synonymy, antonymy, entailment) than the one of ontologies. For this reason, this specific kind of structures have been used for 1) ontology reengineering, as a basis for constructing domain-specific ontologies [26], and 2) as an aid for clustering and comparative analysis of metamodels [27].

D. Semantic Similarity

Semantic similarity deals with the relatedness between concepts, which can be lexicographically different but semantically similar. In this domain, WordNet [28], a thesaurus of the english language, has been the most used lexical database for finding semantic similarity.

Classical algorithms, such as the Leacock-Chodorow, Wu-Palmer, Resnik, or Ling similarities [29], are all based on *IS-A* hierarchical relationships. For instance, Semantic Path Similarity is defined as $path(x, y) = (l(x, y))^{-1}$, with $l(x, y)$ the shortest directed path through a common ancestor.

However, hierarchical approaches have their limitations, as the structure and information content of different ontologies are not directly comparable [30]; furthermore, these approaches do not take advantage of other relationship types of a thesaurus. For these reasons, weight-based, hybrid (supported by an ontology), and feature-based methods are preferred.

In Enterprise Modeling, ArchiMate relationship types can be used to assess semantic similarity, as the language assigns weights to structural relationships. While this weight is employed for applying derivation rules, it is possible to assess the semantic relatedness between concepts of the language.

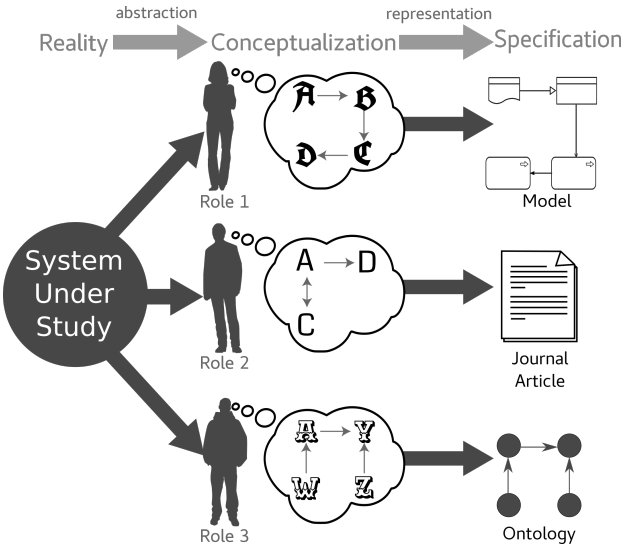

Figure 1. Different conceptualizations among roles.

E. Nuances between Conceptualizations

Ullman's 'meaning triangle' has been widely used to describe the relationship between a **system under study** (SUS), a **conceptualization** of such system, and a **specification**[1] (also called the communicated model or the "language") [31]. Conceptualizations are immaterial entities that *abstract* elements of reality, and only exist in the mind of a person. In order to document, communicate, and analyze these abstractions, specifications are concrete artifacts that *represent* a conceptualization [32].

However, specifications are made by people from different disciplines, backgrounds, levels of expertise and familiarity with the system, and therefore with different conceptualizations of the SUS (see Fig. 1), even between practitioners of the same community. Furthermore, these specifications vary in their degree of formality and detail: some communities use natural language, some use sentential logic, and other use modeling languages or ontologies to communicate their conceptualizations.

These differences stem from the particular world-view and specific needs of each community or individual. In this manner, the *abstraction* process that 'conceptualizes' reality is not the same for everyone, and depends on the intention and requirements of each role. We define **nuances** as differences between conceptualizations, and a *nuanced concept* is a recurring concept between nuances, that despite having different definitions, has the same (or similar) classifier.

These nuances often bring contradictions, not just in the definition of an individual concept, but with regard to related concepts. For instance, Fig. 1 shows three different conceptualizations for the nuanced concept A. In this case, A is directly related to D for *Role 2* ($A \rightarrow D$), whereas for *Role 1* there are some intermediate concepts, i.e. $A \rightarrow B \rightarrow C \rightarrow D$.

[1]To avoid confusion, we refer to specifications, instead of models or languages.

III. A Theory of Nuance

As conceptualizations are immaterial entities in the head of each person, the identification and comparison of nuances, despite being in terms of conceptualizations, must be performed between specifications. However, this has two main issues:

1) Specifications are usually described in different languages, levels of abstraction, and detail.
2) Nuances are not always explicit in specifications, as they contain inherent (i.e. hidden) semantics, as well as contradictions and intermediate concepts.

To overcome the first issue, we can consider the notion of the **rendering** of a specification, which is a translation between specifications. Thus, in order to compare different specifications, we will compare between renderings in the same language. Logical renderings of models [31] and ontologies [32] can be found aplenty in the literature. To overcome the second issue, as well as formalize the definition of nuance, we will employ a particular formal rendering, *Model Theory*, useful to examine the interplay between syntax and semantics.

A. Structures, Languages, and Theories

A **structure** S is an object with four ingredients [33]:

$$S = \{dom(S), P^S, F^S, C^S\} \tag{1}$$

where $dom(S)$, the **domain** of S, is a set of elements, $P^S = \{P_0^S, ..., P_n^S\}$ is a set of **relations**, $F^S = \{F_0^S, ..., F_m^S\}$ is a set of **functions**, and $C^S = \{c_0^S, ..., c_q^S\}$ is a set of **constants**.

A **language** $\mathscr{L}$ [34], also called the *signature* of S [33]:

$$\mathscr{L} = \{P_0, ..., P_n, F_0, ..., F_m, c_0 ..., c_q\} \tag{2}$$

is a collection of symbols, where each P_i is a symbol that names the relation P_i^S, each F_j is a symbol that names the function F_j^S, and each c_k is a symbol that names the constant c_k^S for all $i, j, k \in \mathbb{N}$ such that $i < n$, $j < m$, and $k < q$. This mapping is often called the **interpretation function** $\mathscr{I}$ (cf. [32]) and its inverse, $\mathscr{I}^{-1}$, is the **representation function**.

The symbols of $\mathscr{L}$ can be used to produce **sentences** that describe S. A sentence ϕ is a string of **terms**, where each term is a string of symbols of $\mathscr{L}$, in conjunction with **variables** (i.e. temporary labels for elements of a structure), and the symbols of first-order logic.

A structure S is a model[2] of ϕ (i.e. S *models* ϕ) if ϕ is true in S. This relationship is expressed as: $S \vDash \phi$. A **theory** is a set of ϕ-sentences:

$$\mathscr{T} = \{\phi_0, ..., \phi_n\} \tag{3}$$

such that $S \vDash \phi$ for all $\phi \in \mathscr{T}$.

To illustrate the above definitions in the context of enterprise modeling, we introduce a small modeling language (see Fig. 2), which is a fragment of the ArchiMate language [35].

A rendering of Fig. 2 can be expressed as:

$$S_{mA} = \{dom(mA), P^{mA}, F^{mA}, C^{mA}\}$$

[2]In the strict mathematical sense.

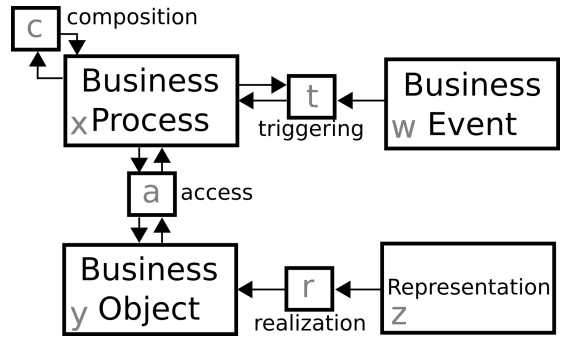

Figure 2. Mini-ArchiMate specification.

$dom(mA)$ is an infinite set of all conceptual entities, and C^{mA} is a set of constants, i.e. specific elements of $dom(mA)$. Edges are expressed with the asymmetric relation R, where mRn holds true iff m and n are connected, for $m, n \in dom(A)$:

$$P^{mA} = \{R\} \quad F^{mA} = \{\emptyset\} \quad C^{mA} = \{w, x, y, z, c, a, r, t\}$$

The language $\mathscr{L}_{mA}$ contains a symbol for R, i.e. $\mathscr{I}^{-1}(R) = \rightarrow$, as well as constants labeling elements of C^{mA}:

$$\mathscr{L}_{mA} = \{\rightarrow, \mathtt{w}, \mathtt{x}, \mathtt{y}, \mathtt{z}, \mathtt{c}, \mathtt{a}, \mathtt{r}, \mathtt{t}\}$$

We differentiate between constants of the domain and constant symbols of the language, e.g. $\mathscr{I}^{-1}(w) = \mathtt{w}$. For brevity, we use the same literal for both; thus, when we use `w`, we are referring to the element labeled `Business Event`, and so on. Finally, we can construct a theory $\mathscr{T}_{mA}$ where $S_{mA} \vDash \mathscr{T}_{mA}$:

$$\mathscr{T}_{mA} = \{\phi\} \qquad \phi = \forall x \neg(x \rightarrow x)$$

This theory just states that no element is connected to itself.

B. Diagrams, Classification, and Substructures

Consider the set of atomic formulas that can be derived from this theory. This set is known as the **diagram** [33] of S_{mA}, expressed as $\mathscr{D}(mA)$. We can find the following 11 sentences among this set, as they hold for ϕ:

$$\begin{array}{llll} \psi_0 = \mathtt{w} \rightarrow \mathtt{t} & \psi_1 = \mathtt{t} \rightarrow \mathtt{x} & \psi_2 = \mathtt{x} \rightarrow \mathtt{t} & \psi_3 = \mathtt{x} \rightarrow \mathtt{c} \\ \psi_4 = \mathtt{c} \rightarrow \mathtt{x} & \psi_5 = \mathtt{x} \rightarrow \mathtt{a} & \psi_6 = \mathtt{a} \rightarrow \mathtt{x} & \psi_7 = \mathtt{a} \rightarrow \mathtt{y} \\ \psi_8 = \mathtt{y} \rightarrow \mathtt{a} & \psi_9 = \mathtt{r} \rightarrow \mathtt{y} & \psi_{10} = \mathtt{z} \rightarrow \mathtt{r} & \end{array} \tag{4}$$

Fig. 2 can be expressed by these sentences, so let us call this subset $\mathscr{D}_e^+(mA) = \{\psi_0, ..., \psi_{10}\}$ the **positive explicit diagram** of S_{mA}. In addition, $\mathscr{D}(mA)$ also contains statements such as:

$$\psi_{11} = \mathtt{t} \rightarrow \mathtt{w} \quad \psi_{12} = \mathtt{r} \rightarrow \mathtt{z} \quad \psi_{13} = \mathtt{z} \rightarrow \mathtt{t} \quad \ldots \tag{5}$$

These statements are valid, but are assumed to be false by their absence in Fig. 2. Thus, let us call the subset $\mathscr{D}_e^-(mA) = \{\neg\psi_{11}, ..., \neg\psi_n\}$ the **negative explicit diagram** of S_{mA} that contains the negation of such statements. The union of both sets is called the **explicit diagram**: $\mathscr{D}_e(mA) = \mathscr{D}_e^+(mA) \cup \mathscr{D}_e^-(mA)$, a finite set that contains either the affirmation or the negation of every possible relation between all $c \in C^{mA}$.

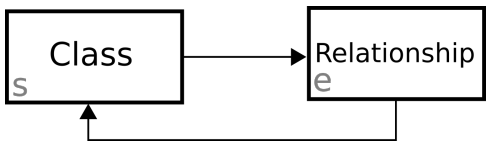

Figure 3. Mini-Ecore specification.

However, $\mathscr{T}_{mA}$ allows other kind of statements, such as:

$$\psi_{n+1} = \mathtt{t} \rightarrow \mathtt{r} \quad \psi_{n+2} = \mathtt{x} \rightarrow \mathtt{w} \quad \psi_{n+3} = \mathtt{z} \rightarrow \mathtt{y} \quad \ldots \quad (6)$$

that do not make much sense in our context, as we would like to differentiate between class elements $(\mathtt{w}, \mathtt{x}, \mathtt{y}, \mathtt{z})$ and relationship elements $(\mathtt{c}, \mathtt{a}, \mathtt{r}, \mathtt{t})$, as described in Fig. 3.

To disallow statements such as the ones of Eq. 6, let $S_{mA'}$ be a structure with the same domain $dom(mA)$, and two additional constants, i.e. $C_{mA'} = C_{mA} \cup \{s, e\}$. This structure has the same set of relations, $P^{mA'} = P^{mA}$, but includes a Γ **classification function**, i.e. $F^{mA'} = \{\Gamma\}$, which is a non-transitive surjective morphism [31] where:

$$\begin{aligned} \Gamma(m) &= \mathtt{s} && \text{for } m \in \{w, x, y, z\} \\ \Gamma(m) &= \mathtt{e} && \text{for } m \in \{c, a, r, t\} \\ \Gamma(m) &= m && \text{for any other element.} \end{aligned}$$

With the language $\mathscr{L}_{mA'} = \mathscr{L}_{mA} \cup \{\mathtt{s}, \mathtt{e}\}$, we can add three sentences to $\mathscr{T}_{mA}$:

$$\begin{aligned} \mathscr{T}_{mA'} &= \mathscr{T}_{mA} \cup \{\phi_0, \phi_1, \phi_2\} \\ \phi_0 &= (\forall m)(\mathtt{s} \rightarrow m \Rightarrow m = \mathtt{e}) \\ \phi_1 &= (\forall m)(\mathtt{e} \rightarrow m \Rightarrow m = \mathtt{s}) \\ \phi_2 &= (\forall m, n)(m \rightarrow n \Rightarrow \Gamma(m) \neq \Gamma(n)) \end{aligned}$$

In this manner, $\mathscr{T}_{mA'}$ renders false formulas such as the ones in Eq. 6. These additions can be expressed as a different structure S_{mM} (see Fig. 3), if we consider Γ, not as a function of a specific structure, but as an **homomorphism** between structures. By the Löwenheim-Skolem theorem [33], S_{mA} and S_{mM} are substructures of $S_{mA'}$. In general, when there is an homomorphism present, it means that we can jump between substructures, and therefore, between specifications.

C. Ambiguous Classification and Hidden Semantics

The partition of substructures described above allows to have incomplete theories that assume other theories, provided an homomorphism between them. For instance, we can consider the specification $\Sigma_A = \{S_{mA}, \mathscr{L}_{mA}, \mathscr{T}_{mA}\}$ as the ArchiMate modeling language, and $\Sigma_M = \{S_{mM}, \mathscr{L}_{mM}, \mathscr{T}_{mM}\}$ as the Ecore metamodel. Given the classification homomorphism Γ that exists between both structures, we can say that the *explicit* ArchiMate specification *assumes* the Ecore specification.

We can formalize this notion as follows: Let a structure S_e be the substructure that generates an **explicit diagram** $\mathscr{D}_e$. We define an **assumed diagram** $\mathscr{D}_a$ as the diagram generated by all the substructures that have an homomorphism with S_e.

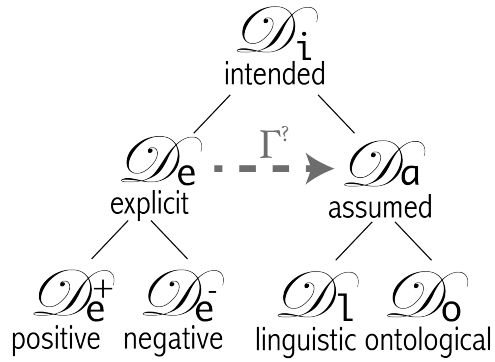

Figure 4. Diagram hierarchy.

Furthermore, we define an **intended diagram** $\mathscr{D}_i = \mathscr{D}_e \cup \mathscr{D}_a$ as the union of the *explicit diagram* and the *assumed diagram*.

A nice property of diagrams is that they can be used to describe structures: For instance, a **positive explicit structure** is a structure that generates $\mathscr{D}_e^+$.

Laarman and Kurtev [36] describe two types of classification, linguistic and ontological, as "*in every language an ontology can be found*". We could differentiate between linguistic (Γ^λ) and ontological classification (Γ^o); however, this distinction is often ambiguous [18]. Thus, we use the symbol $\Gamma^?$ to denote the **ambiguous classification** homomorphism between explicit and assumed structures (see Fig. 4).

We can partition $\mathscr{D}_a$ in two subsets, namely $\mathscr{D}_l$, the **linguistic diagram**, and $\mathscr{D}_o$, the **ontological diagram** (see Fig. 4). We have already outlined the construction process of a linguistic structure (e.g. the substructure S_{mM}), which is a structure that generates $\mathscr{D}_l$. Ontological structures can be constructed in a similar manner, as outlined by Guizzardi [16], [32].

Given a explicit structure S_e, **hidden semantics** is an hypothetical ontological structure S_o^* where an homomorphism between S_e and S_o^* is suggested, but not provided.

D. Nuances

Given two **explicit structures** A and B, with languages $\mathscr{L}_A$ and $\mathscr{L}_B$ respectively, a nuanced concept is present when for a constant symbol $c_A \in \mathscr{L}_A$ there exists a symbol $c_B \in \mathscr{L}_B$ such that:

$$c_A \sigma c_B \quad (7)$$

holds true. The relation σ is a lexical similarity relation (see Sec. II-D) that holds true if c_A is *similar* to c_B. In other words, if two labels are similar, both underlying concepts can be seen as two instances of a nuanced concept.

Formally, a **nuanced concept** $\mathscr{N} = \{c_0, ..., c_n\}$ for a number $n-1$ of explicit structures is a set where $c_i \in dom(S_i)$ for all $0 \leq i < n$ and the following formula holds:

$$(\forall a, b \in \mathscr{N})(\mathscr{I}^{-1}(a) \sigma \mathscr{I}^{-1}(b)) \quad (8)$$

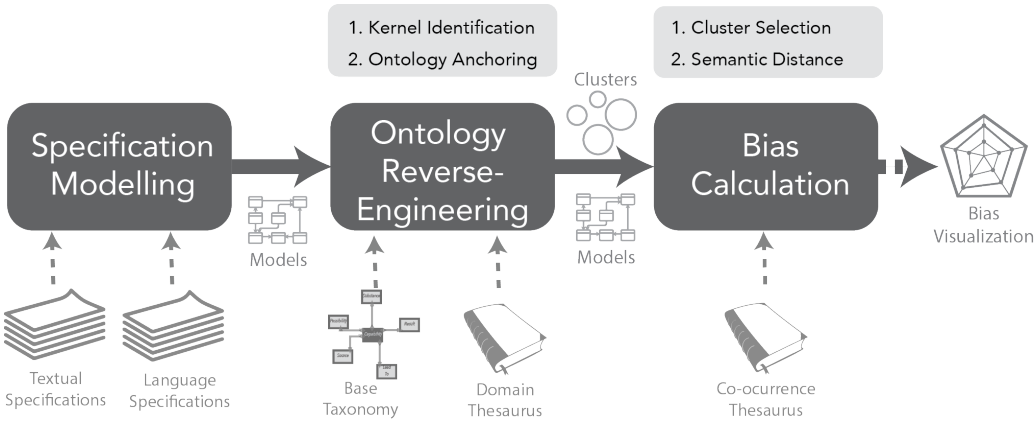

Figure 5. Overview of our Approach.

In particular, the notion of a nuanced concept is of interest when there is not an isomorphism between structures, e.g. when comparing different specifications.

However, nuances are more than just lexical similarity between concepts: we need to obtain other elements that are semantically related to the nuanced concept. Thus, for a structure A that contains a nuanced concept $c \in \mathscr{N}$ with a mapped constant C, i.e. $\mathscr{I}(C) = c$, we define the **nuanced kernel** of C under A:

$$\mathscr{K}_A^C = \{c_0, ..., c_k\} \tag{9}$$

as the constants of A such that the **weighed semantic distance** Δ between c and every $c_i \in \mathscr{K}_A^C$ is less or equal to a maximum semantic distance δ_{max}.

This distance is a function $\Delta : (C^A)^2 \rightarrow \mathbb{R}$ that returns the weighed path distance between $\Gamma^?(c)$ and $\Gamma^?(c_i)$ in an ontological assumed structure. For example, suppose that the constant $\mathtt{x}$ of Fig. 2 is a nuanced concept. Then, in a semantic structure (e.g. a thesaurus) that is assumed for S_{mA}, $\mathtt{z}$ is in the nuanced kernel of $\mathtt{x}$ if $\Delta(\Gamma^?(\mathtt{x}), \Gamma^?(\mathtt{z})) \leq \delta_{max}$ for a given δ_{max}.

Finally, the **bias** of an explicit structure A can be seen as the inclination of a specification towards particular members of $\mathscr{K}_{\mathscr{N}}$, and can be expressed as a vector function:

$$\beta : \Omega_o \rightarrow \mathbb{R}^{|\mathscr{K}_{\mathscr{N}}|} \tag{10}$$

where $|\mathscr{K}_{\mathscr{N}}|$ is the magnitude of the set:

$$\mathscr{K}_{\mathscr{N}} = \{\mathscr{K}_{S_0}^C \cup \cdots \cup \mathscr{K}_{S_n}^C\} \tag{11}$$

In other words, β is an n-dimensional vector, where n is the sum of the magnitude of all nuanced kernels. Each i-coordinate of the vector is the value of $\Delta(\Gamma^?(\mathtt{a}), \Gamma^?(\mathtt{b}))$, where $\mathtt{a} \in \mathscr{N}$ and $\mathtt{b} \in \mathscr{K}_{S_i}^C$.

IV. A Method for Evaluating Bias

In order to analyze the bias of a set of specifications towards a specific interpretation of a nuanced concept, we propose an approach that takes into account semantic relationships between the *nuanced concept* and its *nuanced kernel*.

We will exploit the *ambivalent classification* identified by Atkinson and Kühne [18] and described in the previous section. In this case, we will uncover the *hidden semantics* [19] of such specifications by reverse-engineering an ontology that contains several weighed semantic relationships, which are used to calculate the semantic distance between pairs of concepts. Figure 5 provides an overview of the approach, which is divided in three main stages:

A. Specification Modelling

A key difference between specifications resides in their degree of formality of their renderings: For example, in Enterprise Modeling, some theories of Capabilities come from other disciplines (such as Strategic Management [2], [3], [37]), or even from the private sector (see [38]). These theories lack an explicit metamodel, as they are usually described in knowledge artifacts (e.g. reports, articles, books) written in natural language, with varying granularity and degree of formality. Thus, this first step refers to the construction (or retrieval) of a model that represents a conceptualization of a domain. This model acts as a rendering of the **explicit structure** of Section III-C.

B. Ontology Reverse-Engineering

Reverse-engineering refers to the process of creating representations of a system in another form or at a higher level of abstraction [39]. This stage is concerned with the construction of an ontology (in this case hierarchical taxonomy) for the **nuanced concepts** of the specifications, as well as the related concepts in the **nuanced kernel** for each model.

For this purpose, we perform first a *Kernel Identification*, by calculating the semantic weighed distance Δ. This can be made by using a thesaurus, and calculating the undirected weighed path distance between concepts. Weights are assigned to different similarity relationships (e.g. meronymy, hyponymy, synonymy, entailment).

Table I
THEORIES AND LANGUAGES CHOSEN FOR THIS STUDY.

ID	Type	Domain	Nodes	Rels.	Ref
AM	Language	ArchiMate 3.0	30	31	[35]
i*	Language	i* 2.0	15	77	[40]
CDD	Language	Capability Driven Development	30	118	[5]
BIZ	Text	BIZBOK 2.0	23	34	[41]
CEB	Text	Business Capabilities Handbook	52	140	[38]
LVK	Text	Core Capabilities	27	73	[3]

After that, we create a taxonomy, i.e. hierarchical classification of the elements in the nuanced kernels of all specifications. This structure acts as an *assumed ontological structure*, i.e. an actual structure that represents the *hidden semantics* of all specifications.

Finally, we perform an *Anchoring* [22], which is an ambiguous classification that maps elements in the *reverse-engineered taxonomy* to nuanced concepts of the *specification models*.

C. Bias Calculation

As there can be a large number of semantic dependencies for nuanced elements, first we perform a domain-specific clustering, based on upper levels of the taxonomy constructed in the previous stage. Finally, we calculate the bias as defined at the end of section III-D. We employ a spider chart to visualize the bias of each specification, where each axis represents the weighed semantic distance for a particular cluster.

For instance, if there are eight anchored concepts to a cluster G, and three belong to a specification A, the bias score for this axis is calculated by summing of the weighed semantic distances Δ between the nuanced concept of A and each of the three concepts of the nuanced kernel of A that belong to G. This value is then divided by the sum of all eight deltas, including the ones from other specifications.

V. APPLICATION OF THE METHOD TO BUSINESS CAPABILITIES

Business Capabilities have been recognized as an integral part of Enterprise Architecture nomenclature, and especially a core element of Business Architecture [24]. There is a plethora of bibliography on what a capability is, how it fits on the architecture of the enterprise, and how to model, analyze, and communicate capabilities. Please refer to [7], and [6] for a glimpse on the nuances of this concept, and why this is a problem in Enterprise Modelling. We have selected six specifications: three for DSMLs, and three for textual specifications (see Table I).

In order to evaluate specifications with the same set of tools, we created models for three Enterprise Modelling Languages and three Capability Theories, based on their respective specifications. As the models are quite large for displaying them in this paper, fig. 6 shows the concepts as graph nodes, and the relationships between concepts as edges. For this study, these models were created with the Ecore syntax of the Eclipse Modelling Framework.

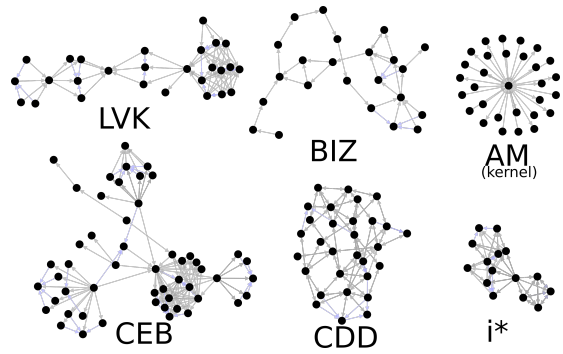

Figure 6. Models selected for this study (see Table I). In the case of ArchiMate, we only present the Capability Kernel, as the entire model is quite large.

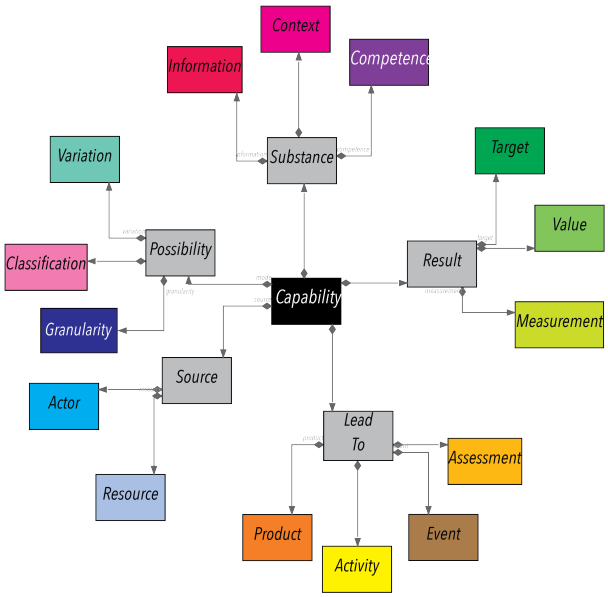

Figure 7. Hierarchical Ontology based on [6] for classifying concepts in Ω_λ.

Given that the CDD, and i* specifications provided the Metamodels, the modelling was straightforward. For BIZ, CEB, and LVK (see Table I), we created linguistic models by modelling our interpretation of each theory.

For five of the models, the kernel was the same as the models for a $\delta_{max} = 8$, as almost all the terms are syntactically connected to the nuanced term Capability. However, the linguistic model of ArchiMate contains 43 non-abstract concepts, and 3337 relationships (by applying ArchiMate derivation rules), resulting in a situation where every concept was related (by some strength) to the Concept of Capability. To obtain the Kernel of ArchiMate, we selected concepts that were directly connected by *Composition*, *Realization*, *Aggregation*, and *Used By* relationship types, and that did not belong to the Technology, Application, and Core domains (see [35]).

A. Ontology Reverse-Engineering

We obtained a *Base Taxonomy* of the concept of Capability, based on the definition of Tell [6]. This definition starts with

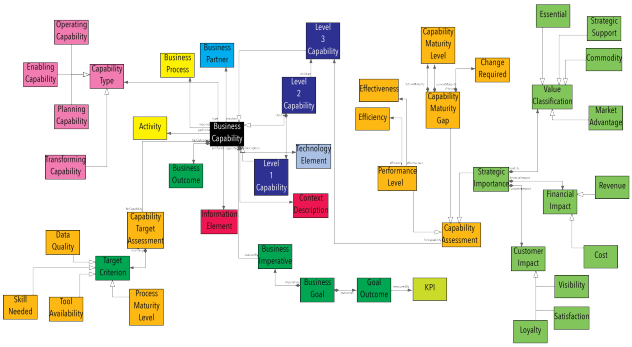

Figure 8. Anchoring of the different concepts in the CEB metamodel to the 15 classifying criteria of Fig. 7.

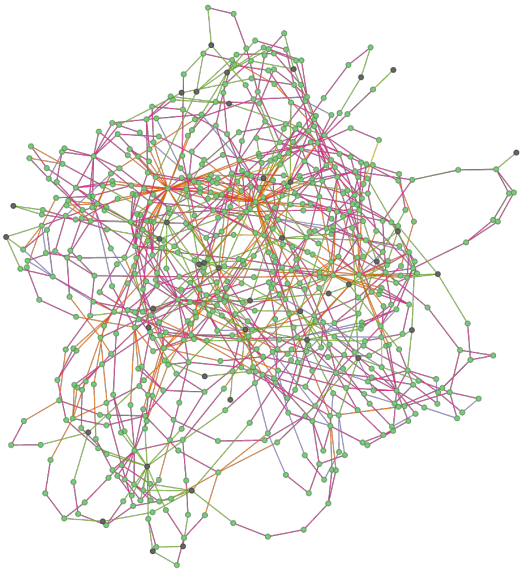

Figure 9. Domain-specific thesaurus for ArchiMate. Colors of relationships represent the type of semantic relationship.

an identification of *what capability is not*, and identifying the essential elements of any definition of capability. This is the first level of the taxonomy, which represents essential categories for the concept (see Fig. 7). Then, we expanded the ontology with sub-categories for each category described by Tell. Finally, we anchored the concepts of each specification with categories on the base taxonomy. Fig. 8 shows the anchoring of the CEB specification: Each color represents an anchoring of the concept to a sub-category of Fig. 7.

For the automatic anchoring, we created a Thesaurus of each μ_λ, using the English corpus of WordNet [28]. In contrast with [27], we calculated similarity using eight relationship types of the taxonomy. This is because, as suggested by Deissenboeck and Ratiu [39], we have to consider several semantic similarity relationship types when performing semantic reverse engineering of a domain.

B. Bias Calculation

Given the 15 clusters of the base ontology (which represent the nuances of the Capability concept), as well as the anchored concepts for each cluster, we proceeded to calculate the bias of each specification towards each nuance. For obtaining the coordinates of the 15-dimensional vector β (see Eq. 10), we generated a special semantic structure (see Fig. 9) that, in contrast to general thesaurus, it only contains the shortest paths between the dependencies of the nuanced concepts (see Fig. 10 for an example with two concepts).

To create this structure, we analyzed the WordNet thesaurus [28], and extracted Synsets that were in shortest paths of maximum length $\delta_{max} = 8$. Then, we assigned weights to 8 different semantic similarity relationships: *hyponym*, *hypernym*, *category*, *member meronym*, *similar*, *member of category domain*, *instances hyponym*, and *substance meronym*. This allows a straightforward calculation of Δ, which is the length of the path between the nuanced concept C (i.e. *Capability*) and each c_i member of the kernel $\mathscr{K}_A^C$. The score for a nuance in a specification A is calculated with Eq. 12, where k is the size of the kernel, and $m = max(\Delta(C, c_i))$ for all $c_i \in \mathscr{K}_A^C$:

$$score(\mathscr{K}_A^C) = \sum_{i=1}^{k} \frac{1}{k} * \frac{1}{\Delta(C, c_i) * (m)} \tag{12}$$

Results of the evaluation can be seen in Table II, where N is the total number of concepts for each nuance. These results are visualized in Fig. 11 as spider charts, where it is possible to infer the distribution of each nuance (the radial axis) for the six specifications. Furthermore, the **bias** of each specification can be easily inferred: For instance, if we were interested in a theory that describes the *Contextual*, *Variation*, and *Event* aspects of a Capability, i.e. we are biased towards a particular interpretation of Capability, we would certainly choose the CDD theory and metamodel, instead of e.g. the BIZBOK or i* theories, which do not have concepts for such aspects. On the other hand, if we were interested on more general theories of Capability, we would choose ArchiMate, which contains several common aspects of the concept. We could even use both ArchiMate and CEB, as their combination covers most nuances (all except Classification, Measurement, and Variation).

Table II
WEIGHED DISTRIBUTION OF NUANCES AMONG SPECIFICATIONS

ID	Nuance	N	AM	BIZ	CDD	CEB	i*	LVK
A	Activity	7	28.5%	0%	14.2%	38.1%	0%	14.3%
B	Actor	15	13.3%	23.8%	0%	6.7%	22.7%	0%
C	Assessment	38	19.4%	18.3%	0%	56.7%	7.9%	12.3%
D	Classification	12	0%	0%	0%	43.1%	0%	76.4%
E	Competence	3	0%	0%	0%	0%	0%	100%
F	Context	11	21.8%	0%	47.3%	9.1%	9.1%	9.1%
G	Event	2	50%	0%	50%	0%	0%	0%
H	Granularity	3	0%	0%	0%	100%	0%	0%
I	Information	5	66.7%	0%	0%	20%	0%	20%
J	Measurement	10	0%	0%	64.8%	10%	0%	0%
K	Product	9	26.7%	52.6%	0%	0%	0%	0%
L	Resource	5	44%	0%	20%	20%	20%	0%
M	Target	17	12.7%	24.4%	5.9%	35.3%	5.9%	11.8%
N	Value	20	5%	10.8%	0%	61.1%	0%	23.5%
O	Variation	10	0%	0%	100%	0%	0%	0%

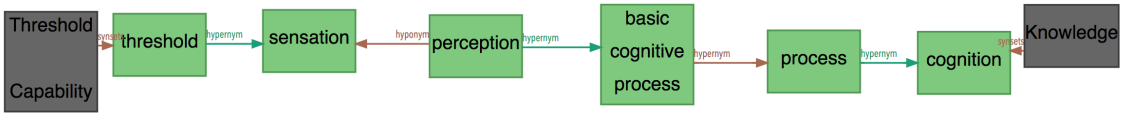

Figure 10. Example of a thesaurus path of $\Delta = 6$ between the concepts `Threshold Capability` and `Knowledge`, found in the LVK specification.

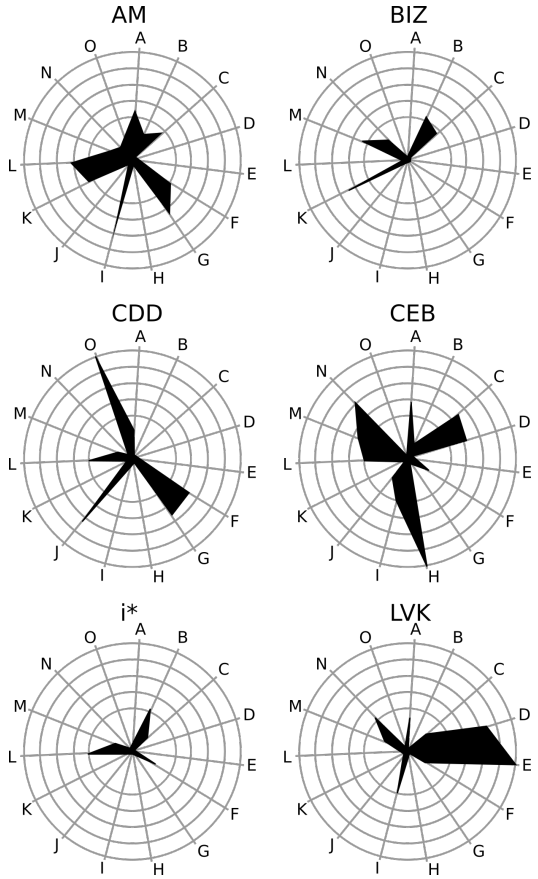

Figure 11. Evaluation of the Bias of each Specification towards 15 nuances found in the corpus. Please refer to Table II for the label of each axis.

C. Related Work

The notions of interpretation and meanings has been explored in Intensional Logics (e.g. Propositional Modal Logic), a formal approach to *intensions*, which represent meanings, as opposed to *extensions*, which represent designators. These logics are based on Carnap's idea that intensions can be given a precise mathematical embodiment as functions on states. However, Intensional Logics do not allow the precise definition of Nuances, which we consider an important construct in the comparison of theories. Instead, we employ Model Theory, which makes the distinction between Languages, Theories, and Structures, allowing a formal definition of nuance.

Karagiannis et al. [19] approach the hidden semantics of metamodels by anchoring domain-specific ontologies that enrich the semantics of modeling tools. Guizzardi et al. [16], [32], [43] provide a large body of work for the support of modeling language design, and introduce the Unified Foundational Ontology (UFO) that provides an isomorphism between ontologies and modeling languages, thus supporting the construction of languages that are ontologically sound. These approaches, in conjunction with the work of Atkinson and Kühne [18], are used to provide a conceptual foundation for a theory of nuance.

Deissenboeck and Ratiu [39] approach the semantic reverse-engineering of source code artifacts, in order to uncover the semantics of compiled programs. We adopt this notion of semantic reverse-engineering, and apply it to a set of models.

Recently, Pittke et al. [42] proposed an approach for the detection and resolution of lexical ambiguity in Process Models. Their approach uses the BabelNet[3] corpus for Word Sense Disambiguation by identifying homonyms and synonyms in the designators of process activities. While their scope is more specific, we consider that their approach can be used in our framework as the Δ function that returns the semantic distance between concepts.

In [44] and [27], Babur et al. explore the joint analysis of a corpus of models by constructing an unified model. In this study, hierarchical IS-A relationships (hyponymy and hypernymy) are used to merge similarities between concepts, i.e. by finding clusters that contain similar labels. While this approach is valuable for performing specification comparison, it only describes similarities in terms of domain-agnostic clusters, as this process is automated. Therefore, we employ a mixed approach, where the classifying ontology is created systematically, e.g. by the negative approach of Tell [6]. The advantage of using a mixed approach is that we can obtain domain-specific clusters that provide a more descriptive ontology of the domain, allowing the identification of nuances.

VI. Conclusion and Outlook

In this paper, we examined the relationship between syntax and semantics, and took advantage of ambivalent classification to uncover the hidden semantics of specifications. Then, we proposed an approach for calculating the bias of these specifications towards particular nuances. The application of this approach can help in the comparison between theories, languages, and metamodels, in order to select the best one for modeling a domain of a particular enterprise. In addition, it can be used to merge different specifications, with the aim of having a more complete metamodel of a domain.

[3] A lexical database based in WordNet, enriched with encyclopedic knowledge from Wikipedia [42].

The accuracy of the evaluation depends on five qualities: 1) The accuracy of the models created from specifications, 2) How many specifications are evaluated, 3) The relationship types selected for linguistic matching, 4) The ontological strength of the taxonomy, and 5) The selected algorithm for semantic distance calculation. Thus, future work is directed on improving these five qualities. For instance, Machine Learning approaches can be employed to support the semi-automatic creation of the Base Ontology, and different approaches, e.g. [42] can be used to calculate the semantic distance.

Acknowledgment

This work was supported by the COLCIENCIAS grant 727 for doctoral studies.

References

[1] B. Selic, "The pragmatics of model-driven development," *IEEE Software*, vol. 20, no. 5, pp. 19–25, Sep. 2003.

[2] H. I. Ansoff, *Corporate strategy: An analytic approach to business policy for growth and expansion*. McGraw-Hill Companies, 1965.

[3] C. Long and M. Vickers-Koch, "Using core capabilities to create competitive advantage," *Organizational dynamics*, vol. 24, no. 1, pp. 7–22, 1995.

[4] D. Beimborn, S. F. Martin, and U. Homann, "Capability-oriented modeling of the firm," in *IPSI Conference*, 2005.

[5] S. Bērziša, G. Bravos, T. C. Gonzalez, U. Czubayko, S. España, J. Grabis, M. Henkel, L. Jokste, J. Kampars, H. Koç, J.-C. Kuhr, C. Llorca, P. Loucopoulos, R. J. Pascual, O. Pastor, K. Sandkuhl, H. Simic, J. Stirna, F. G. Valverde, and J. Zdravkovic, "Capability Driven Development: An Approach to Designing Digital Enterprises," *Business & Information Systems Engineering*, vol. 57, no. 1, pp. 15–25, Feb. 2015.

[6] A. W. Tell, "What capability is not," in *International Conference on Business Informatics Research*. Springer, 2014, pp. 128–142. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-319-11370-8_10

[7] G. Dosi, R. Nelson, and S. Winter, *The nature and dynamics of organizational capabilities*. OUP Oxford, 2001.

[8] J. Ludewig, "Models in software engineering - an introduction," *Software and Systems Modeling*, vol. 2, no. 1, pp. 5–14, Mar. 2003.

[9] P.-A. Muller, F. Fondement, B. Baudry, and B. Combemale, "Modeling modeling modeling," *Software & Systems Modeling*, vol. 11, no. 3, pp. 347–359, Jul. 2012.

[10] E. Rodriguez-Priego, F. J. García-Izquierdo, and Á. L. Rubio, "References-enriched Concept Map: a tool for collecting and comparing disparate definitions appearing in multiple references," *Journal of Information Science*, vol. 39, no. 6, pp. 789–804, Dec. 2013.

[11] C. W. Morris, *Writings on the general theory of signs*. Walter de Gruyter, 1971, vol. 16.

[12] E. Seidewitz, "What models mean," *IEEE software*, vol. 20, no. 5, pp. 26–32, 2003.

[13] T. Kühne, "Matters of (meta-) modeling," *Software & Systems Modeling*, vol. 5, no. 4, pp. 369–385, 2006.

[14] H. Stachowiak, *Allgemeine Modelltheorie*. Springer-Verlag, 1973.

[15] D. Harel and B. Rumpe, "Meaningful modeling: what's the semantics of" semantics"?" *Computer*, vol. 37, no. 10, pp. 64–72, 2004.

[16] G. Guizzardi, *Ontological foundations for structural conceptual models*. Enschede : Enschede: University of Twente. Centre for telematics and information technology ; Telematics instituut, 2005, oCLC: 799205844.

[17] T. Kühne, "Clarifying matters of (meta-) modeling: an author's reply," *Software & Systems Modeling*, vol. 5, no. 4, pp. 395–401, Dec. 2006.

[18] C. Atkinson and T. Kühne, "Demystifying Ontological Classification in Language Engineering," in *Modelling Foundations and Applications: 12th European Conference, ECMFA 2016, Held as Part of STAF 2016, Vienna, Austria, July 6-7, 2016, Proceedings*. Cham: Springer International Publishing, 2016, pp. 83–100.

[19] D. Karagiannis and P. Höfferer, "Metamodels in action: An overview," in *ICSOFT 2006, First International Conference on Software and Data Technologies, Setúbal, Portugal, September 11-14, 2006*, J. Filipe, B. Shishkov, and M. Helfert, Eds. INSTICC Press, 2006.

[20] A. Third, "Hidden semantics: what can we learn from the names in an ontology?" in *Proceedings of the Seventh International Natural Language Generation Conference*. Association for Computational Linguistics, 2012, pp. 67–75.

[21] M.-N. Terrasse, M. Savonnet, E. Leclercq, T. Grison, and G. Becker, "Do We Need Metamodels AND Ontologies for Engineering Platforms?" in *Proceedings of the 2006 International Workshop on Global Integrated Model Management*, ser. GaMMa '06. New York, NY, USA: ACM, 2006, pp. 21–28.

[22] P. Höfferer, "Achieving Business Process Model Interoperability Using Metamodels and Ontologies." in *ECIS*, 2007, pp. 1620–1631.

[23] A. M. Sutti, T. Verhoeff, and M. G. J. van den Brand, "Ontologies in domain specific languages: a systematic literature review," Technische Universiteit Eindhoven, Eindhoven, Tech. Rep., 2014.

[24] T. O. Group, "Business Capabilities," Tech. Rep., 2016.

[25] N. Grabar, T. Hamon, and O. Bodenreider, "Ontologies and terminologies: Continuum or dichotomy?" *Applied ontology*, vol. 7, no. 4, pp. 375–386, 2012.

[26] D. Kless, S. Milton, and E. Kazmierczak, "Relationships and relata in ontologies and thesauri: Differences and similarities," *Applied Ontology*, vol. 7, no. 4, pp. 401–428, Jan. 2012.

[27] Ö. Babur, L. Cleophas, and M. v. d. Brand, "Hierarchical Clustering of Metamodels for Comparative Analysis and Visualization," in *Modelling Foundations and Applications*, ser. Lecture Notes in Computer Science. Springer International Publishing, Jul. 2016, pp. 3–18.

[28] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[29] E. Deza and M. M. Deza, *Encyclopedia of Distances*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, dOI: 10.1007/978-3-642-00234-2.

[30] G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. Petrakis, and E. E. Milios, "Semantic Similarity Methods in wordNet and Their Application to Information Retrieval on the Web," in *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management*, ser. WIDM '05. New York, NY, USA: ACM, 2005, pp. 10–16.

[31] B. Henderson-Sellers, *On the Mathematics of Modelling, Metamodelling, Ontologies and Modelling Languages*, ser. SpringerBriefs in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, dOI: 10.1007/978-3-642-29825-7.

[32] G. Guizzardi, "On Ontology, Ontologies, Conceptualizations, Modeling Languages, and (Meta)Models," in *Proceedings of the 2007 Conference on Databases and Information Systems IV: Selected Papers from the Seventh International Baltic Conference DB&IS'2006*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2007, pp. 18–39.

[33] W. Hodges, *A Shorter Model Theory*. New York, NY, USA: Cambridge University Press, 1997.

[34] C. Chang and H. Keisler, *Model Theory*, 3rd ed., ser. Studies in Logic and the Foundations of Mathematics. Elsevier, Academic Press, 1990.

[35] The Open Group, "ArchiMate ® 3.0 Specification," Tech. Rep., 2016.

[36] A. Laarman and I. Kurtev, "Ontological Metamodeling with Explicit Instantiation," in *Software Language Engineering*. Springer, Berlin, Heidelberg, Oct. 2009, pp. 174–183.

[37] D. Leonard-Barton, "Core capabilities and core rigidities: A paradox in managing new product development," *Strategic management journal*, vol. 13, no. S1, pp. 111–125, 1992.

[38] CEB, "The Architect's Business Capabilities Handbook," 2013.

[39] F. Deissenboeck and D. Ratiu, "A unified meta-model for concept-based reverse engineering," in *Proceedings of the 3rd International Workshop on Metamodels, Schemas, Grammars and Ontologies (ATEM'06)*, 2006.

[40] F. Dalpiaz, X. Franch, and J. Horkoff, "istar 2.0 language guide," *arXiv preprint arXiv:1605.07767*, 2016.

[41] B. A. Guild, "A Guide to the Business Architecture Body of Knowledge (BIZBOK Guide), V. 4.5," Tech. Rep., 2015. [Online]. Available: http://www.businessarchitectureguild.org/

[42] F. Pittke, H. Leopold, and J. Mendling, "Automatic Detection and Resolution of Lexical Ambiguity in Process Models," *IEEE Transactions on Software Engineering*, vol. 41, no. 6, pp. 526–544, Jun. 2015.

[43] G. Guizzardi and G. Wagner, "A Unified Foundational Ontology and some Applications of it in Business Modeling." in *CAiSE Workshops (3)*, 2004, pp. 129–143.

[44] Ö. Babur, L. Cleophas, T. Verhoeff, and M. van den Brand, "Towards Statistical Comparison and Analysis of Models:." SCITEPRESS - Science and and Technology Publications, 2016, pp. 361–367.