

Ontology-Based Evaluation and Design of Visual Conceptual Modeling Languages

Giancarlo Guizzardi

Ontology and Conceptual Modeling Research Group (NEMO),
Federal University of Espírito Santo (UFES), Vitoria, Brazil

Abstract In this chapter, we present a framework for the evaluation and (re)design of modeling languages. In our approach, this property can be systematically evaluated by comparing a concrete representation of the worldview underlying the language (captured in the language's meta-model), with an explicit and formal representation of a conceptualization of that domain (a reference ontology). Moreover, we elaborate on formal characterizations for the notions of reference ontology, conceptualization and meta-model, as well as on the relations between them. By doing this, we can also formally define the relation between the state of affairs in reality deemed possible by an ontology and the grammatical models admitted by a modeling language. The precise characterization of this relation allows for a systematic improvement of a modeling language by incorporating ontological axioms as grammatical constraints in the language's meta-model. Furthermore, we demonstrate how an approach based on visual simulation could be used to assess this relation, i.e., to evaluate the distance between the *valid models* of a language and the *intended models* according to the underlying conceptualization. Finally, we demonstrate how the use of a system of formal ontological properties can be systematically exploited in the design of pragmatically efficient Domain-Specific Visual Languages.

Keywords: Ontology, Visual Languages, Language Engineering

1 Introduction

The objective of this chapter is to discuss the design and evaluation of modeling languages for capturing phenomena in a given domain according to a conceptualization of that domain. In particular, we focus on two properties of a modeling language with respect to a given real-world domain [1]: (i) *domain appropriateness*, which refers to truthfulness of the language to the domain and (ii) *comprehensibility appropriateness*, which refers to the pragmatic efficiency of the language to support communication, understanding and reasoning in the domain.

The elements constituting a *conceptualization* of a given domain are used to articulate abstractions of certain state of affairs in reality. We name them here *do-*

main abstractions. Domain conceptualizations and abstractions are intangible entities that only exist in the mind of the user or a community of users of a language. In order to be documented, communicated and analyzed, these entities must be captured in terms of some concrete artifact, namely a *model*. Moreover, in order to represent a model, a *modeling language* is necessary. Figure 1 depicts the relation between a conceptualization, domain abstraction, model and modeling language.

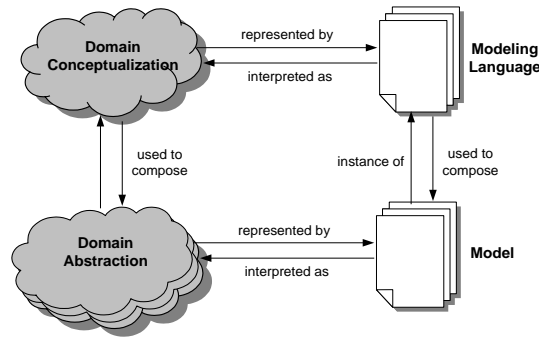


Fig. 1. Relations between conceptualization, abstraction, modeling language and model.

In this chapter, we elaborate on a framework that can be used to evaluate the suitability of a language to model a set of real-world phenomena in a given domain. In our approach, *domain* and *comprehensibility appropriateness* can be systematically evaluated by comparing the level of homomorphism between a concrete representation of the worldview underlying the language (captured in a *meta-model of the language*), with an explicit and formal representation of a conceptualization of that domain (a *reference ontology* [2]). Our framework comprises a number of properties that must be reinforced for an isomorphism to take place between these two entities. If an isomorphism can be guaranteed, the implication for the human agent who interprets a diagram (model) is that his interpretation correlates precisely and uniquely with an abstraction being represented. By contrast, in case the correlation is not an isomorphism there may be multiple unintended abstractions that match the interpretation.

The framework presented here builds on existing work in the literature. In particular, it considers the frameworks proposed in [3,4], which focus on evaluating the match between individual diagrams and the state of affairs they represent, and the pioneering approach of Wand and Weber presented in [5,6], which focuses on the system of representations as a whole, i.e., a language. Although our approach is also centered in the language level, we show that, by considering desirable properties of the mapping of individual diagrams onto what they represent, we are able to account for desirable properties of the diagrams' modeling languages. In this way, we extend the original proposal presented in [5]. We also build here on the work of the philosopher of language H. P. Grice [7] and his notion of *conversational maxims* that states that a speaker is assumed to make in dialogue contributions which are *relevant*, *clear*, *unambiguous*, and *brief*, *not overly informative*

and *true according to the speaker's knowledge*. Furthermore, in comparison to [3,4] and [5], by presenting a formal elaboration of the nature of the entities depicted in Figure 1 as well as their interrelationships, we manage to present a more general and precise characterization of the characteristics that a language must have to be considered truthful to a given domain.

The rest of this chapter is structured as follows. Section 2 introduces the evaluation framework proposed here. Section 3 presents a formal characterization of the notions of reference ontology, conceptualization and meta-model, as well as on the relations between these notions. By doing this, we can also formally define the relation between the state of affairs in reality deemed possible by a reference ontology and the grammatical models admitted by a modeling language. In section 4, we exemplify the approach proposed by reporting on the design of an Ontologically Well-Founded version of UML for the purpose of Conceptual Modeling and Domain Ontology Engineering. This language (now termed *OntoUML*), in addition to being an extensive case study of the approach discussed here, is itself a contribution to the engineering of Domain-Specific Languages. This is discussed in depth in section 5. Finally, section 6 presents final considerations of the chapter.

It is important to highlight that this chapter can be considered as an extension of [1]. In particular, sections 4 and 5 represent a substantial extension to the original paper. Section 6 also contains a more systematic comparison with the works of Gurr and Wand & Weber.

2 Language and Conceptualization

The purpose of the current chapter is to discuss the design and evaluation of artificial modeling languages for capturing phenomena in a given material domain according to a conceptualization of this domain. Before targeting this at a language level, i.e., at a level of a system of representations, we start discussing the simpler relation between particular models and abstractions of portions of reality.

In [3,4], Gurr presents a framework to formally evaluate the relation between the properties of a representation system and the properties of the domain entities they represent. According to him, representations are more or less effective depending on the level of homomorphism between the algebras used to represent what he terms the *representing* and the *represented* world, which correspond to the *model* and *domain abstraction* in figure 1, respectively.

Gurr argues at length that the stronger the match between a model and its representing diagram, the easier it is to reason with the latter. The easiest case is when these matches are *isomorphisms*. The implication of this for the human agent who interprets the diagram is that his interpretation correlates precisely and uniquely with an abstraction being represented. By contrast, where the correlation is not an isomorphism then there may potentially be a number of different models that would match the interpretation.

The evaluation framework proposed by Gurr focuses on evaluating the match between individual diagrams and the state of affairs (*abstractions*) they represent.

In [5,6], another framework is defined for evaluating *expressiveness* and *clarity* of modeling grammars, i.e., with the focus on the system of representations as a whole. In other words, in the latter proposal, the authors focus on the relation between what is named *Conceptualization* and *Modeling Language* in figure 1. In this chapter, these two proposals are merged into one single evaluation framework. We focus our evaluation on the level of the system of representations. Nevertheless, as it will be shown in the following subsections, by considering desirable properties of the mapping of individual diagrams onto what they represent, we are able to account for desirable properties of the modeling languages used to produce these diagrams, extending in this way Wand & Weber's original proposal.

It is important to highlight that in the proposal discussed here we want to systematically evaluate the level of homomorphism between *Conceptualization* and *Language* by comparing concrete representation of these entities: the notion of an *Ontology* as a concrete representation of a conceptualization is discussed in depth and formally characterized in section 3; as a concrete representation of a language, we take the language meta-model. It is important to clarify, nonetheless, that by meta-model of the language we do not mean the actual description of the abstract syntax of the language. Instead what is meant here is what is termed in [2] the *Ontological Meta-model of the Language* or, simply, the *Ontology of the Language*. This meta-model is meant to capture the worldview underlying the language represented by the language modeling primitives. The definitive abstract syntax of the language is a language engineering artifact derived from that by considering a number of relevant non-functional requirements (e.g., to facilitate meta-model management or mapping to a particular implementation technology, decidability and complexity in reasoning, etc.) [2].

In [3], four properties are defined, which are required to hold for a homomorphic correlation between a *represented world* and a *representation* to be an isomorphism: *lucidity*, *soundness*, *laconicity* and *completeness* (figure 2). These properties are discussed as follows.

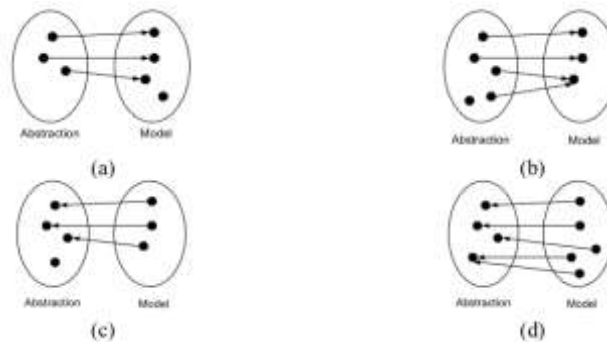


Fig. 2. Examples of Lucid (a) and Sound (b) representational mappings from Abstraction to Model; Examples of Laconic (c) and Complete (d) interpretation mappings from Model to Abstraction

2.1 Lucidity and Construct Overload

A model \mathcal{M} is called *lucid* with respect to (w.r.t.) an abstraction \mathcal{A} if a (representation) mapping from \mathcal{A} to \mathcal{M} is *injective*. A mapping between \mathcal{A} and \mathcal{M} is injective iff every entity in the model \mathcal{M} represents *at most* one (although perhaps none) entity of the abstraction \mathcal{A} . An example of an injective mapping is depicted in figure 2.(a).

The notion of lucidity at the level of individual diagrams is strongly related to the notion of *ontological clarity* at the language level as discussed in [6]. In that article, the author states that the ontological clarity of a modeling grammar is undermined by what he calls *construct overload*: “*construct overload occurs when a single grammatical construct can stand for two or more ontological constructs. The grammatical construct is overloaded because it is being used to do more than one job.*”

The notions of *lucidity* and *ontological clarity* albeit related are not identical. A construct can be overloaded in the language level, i.e., it can be used to represent different concepts, but every manifestation of this construct in individual specifications is used to represent only one of the possible concepts. Nevertheless, non-lucidity can also be manifested at a language level. We say that a language (system of representation) is non-lucid according to a conceptualization if there is a construct of the language which is non-lucid, i.e., a construct that when used in a model stands for more than one entity of the represented abstraction. Non-lucidity at the language level is a special case of construct overload that does entail non-lucidity at the level of individual specifications.

Construct overload is considered an undesirable property of a modeling language since it causes ambiguity and, hence, undermines clarity. When it exists, users have to bring additional knowledge not contained in the specification to understand the phenomena that are being represented. In summary, a modeling language should not contain construct overload and every instance of a modeling construct of this language should represent only one individual of the represented domain abstraction.

2.2 Soundness and Construct Excess

A model \mathcal{M} is called *sound* w.r.t. an abstraction \mathcal{A} if a (representation) mapping from \mathcal{A} to \mathcal{M} is *surjective*. A representation mapping from \mathcal{A} to \mathcal{M} is surjective iff the corresponding interpretation mapping from \mathcal{M} and \mathcal{A} is total, i.e., iff every entity in the model \mathcal{M} represents *at least* one entity of abstraction \mathcal{A} (although perhaps several). An example of a surjective representation mapping is depicted in figure 2.(b).

Unsoundness at the level of individual specifications is strongly related to unsoundness at the language level, a property that is termed *construct excess* in [6]:

“*construct excess occurs when a grammatical construct does not map onto an ontological construct*”. Although construct excess can result in the creation of unsound specifications, soundness at the language level does not prohibit the creation of unsound specifications. For instance, suppose a domain of natural numbers and a language that uses arrows to represent the *less-than* relation between natural numbers and labeled boxes to represent these numbers. Now, suppose we use this language to build a specification in which we have a box labeled X *arrow-connected* to the box representing the number 0. Although the language used is sound, i.e., all construct types have an interpretation in terms of domain types, the aforementioned specification produced using the language is unsound, given that there is no referent to the box labeled X in the domain. Since no mapping is defined for the exceeding construct, its meaning becomes uncertain, hence, undermining the clarity of the specification.

According to [6], users of a modeling language must be able to make a clear link between a modeling construct and its interpretation in terms of domain concepts. Otherwise, they will be unable to articulate precisely the meaning of the specifications they generate using the language. Therefore, a modeling language should not contain construct excess and every instance of its modeling constructs must represent an individual in the domain.

2.3 Laconicity and Construct Redundancy

A model \mathcal{M} is called *laconic* w.r.t. an abstraction \mathcal{A} if the interpretation mapping from \mathcal{M} to \mathcal{A} is *injective*, i.e., iff every entity in the abstraction \mathcal{A} is represented by **at most** one (although perhaps none) entity in the model \mathcal{M} . An example of an injective interpretation mapping is depicted in figure 2.(c). The notion of *laconicity* in the level of individual specifications is related to the notion of **construct redundancy** in the language level in [6]: “*construct redundancy occurs when more than one grammatical construct can be used to represent the same ontological construct.*”

Once again, despite of being related, laconicity and construct redundancy are two different (even opposite) notions. On one hand, construct redundancy does not entail non-laconicity. For example, a language can have two different constructs to represent the same concept. However, in every situation the construct is used in particular specifications, it only represents a single domain element. On the other hand, the lack of construct redundancy in a language does not prevent the creation of non-laconic specifications in that language. For example, the *arrow/labeled box* language for representing natural numbers in section 2.2 is laconic, i.e., for each domain type there is at most one construct type in the language. However, we can still produce using this simple language a specification in which, for example, the same natural number (e.g., 3) is represented by more than one labeled box.

Non-laconicity can also be manifested at the language level. We say that a language is non-laconic if it has a non-laconic modeling construct, i.e., a construct

that when used in a specification of a model causes an entity of this model to be represented more than once. Non-laconicity at the language level is a special case of construct redundancy that does entail non-laconicity at the level of individual diagrams.

In [6], the authors claim that construct redundancy “*adds unnecessarily to the complexity of the modeling language*” and that “*unless users have in-depth knowledge of the grammar, they may be confused by the redundant construct. They might assume for example that the construct somehow stands for some other type of phenomenon.*” Therefore, construct redundancy can also be considered to undermine representation clarity. In summary, a modeling language should not contain construct redundancy, and elements in the represented domain should be represented by at most one instance of the language modeling constructs.

2.4 Completeness

A model \mathcal{M} is called *complete* w.r.t. an abstraction \mathcal{A} if an interpretation mapping from \mathcal{M} to \mathcal{A} is *surjective*. An interpretation mapping from \mathcal{M} to \mathcal{A} is surjective iff the corresponding representation mapping from \mathcal{A} to \mathcal{M} is total, i.e., iff every entity in an abstraction \mathcal{A} (instance of the domain conceptualization) is represented by **at least** one (although perhaps many) entity in the model \mathcal{M} . An example of a *surjective* interpretation mapping is depicted in figure 2.(d).

The notion of completeness at the level of individual specifications is related to the notion of *ontological expressiveness* and, more specifically, *completeness* at the language level, which is perhaps the most important property that should hold for a representation system. A modeling language is said to be *complete* if every concept in a domain conceptualization is covered by at least one modeling construct of the language. Language incompleteness (also termed *Construct Deficit*) entails lack of expressivity, i.e., that there are phenomena in the considered domain (according to a conceptualization) that cannot be represented by the language. Alternatively, users of the language can choose to overload an existing construct, thus, undermining clarity.

An incomplete modeling language is bound to produce incomplete specifications unless some existing construct is overloaded. However, the converse is not true, i.e., a complete language can still be used to produce incomplete specifications. Once more, we refer to the arrow/labeled box language of previous sections. This language is complete, i.e., for each domain type there is at least one construct type in the language. However, we can still produce using this language a specification in which, for example, a relation instance is missing (e.g., the less-than relation between the boxes representing numbers 2 and 3).

3 Conceptualization, Ontology and Meta-model

Let us now return our attention to figure 1. A modeling language can be seen as delimiting all possible specifications¹ which can be constructed using that language, i.e., all grammatically valid specifications of that language. Likewise, a conceptualization can be seen as delimiting all possible domain abstractions (representing state of affairs) which are admissible in that domain [8]. Therefore, for example, in a conceptualization of the domain of genealogy, there cannot be a domain abstraction in which a person is his own biological parent, because such a state of affairs cannot happen in reality. Accordingly, we can say that a modeling language that is truthful to this domain is one that has as valid (i.e., grammatically correct) specifications only those that represent state of affairs deemed admissible by a conceptualization of that domain. In the sequel, we review a formalization of this idea presented at [2], which is an extension of the original idea proposed in [8]. This formalization compares conceptualizations as *intentional structures* and meta-models as represented by logical theories. Thus, in the sequel, we make use of the terms *possible world*, *domain of quantification*, *relation* and *interpretation function* in their traditional established sense in model logics [9].

Let us first define a *conceptualization* C as an intensional structure $\langle W, D, \mathfrak{R} \rangle$ such that W is a (non-empty) set of possible worlds, D is the domain of individuals and \mathfrak{R} is the set of relations (concepts) that are considered in C . The elements $\rho \in \mathfrak{R}$ are intensional (or conceptual) relations with signatures such as $\rho^n: W \rightarrow \wp(D^n)$, such that n is the arity of ρ , and so that each relation is a function from possible worlds to sets of n -tuples of individuals in the domain. For instance, we can have ρ accounting for the meaning of the natural kind *Apple*. In this case, the meaning of *Apple* is captured by the intentional function ρ , which refers to all instances of *Apple* in every possible world. For every world $w \in W$, according to C we have an *intended world structure* $S_w C$ as a structure $\langle D, R_w C \rangle$ such that $R_w C = \{\rho(w) \mid \rho \in \mathfrak{R}\}$. More informally, we can say that every intended world structure $S_w C$ is the characterization of some state of affairs in world w deemed admissible by conceptualization C . From a complementary perspective, C defines all the admissible state of affairs in that domain, which are represented by the set $S_c = \{S_w C \mid w \in W\}$.

Let us consider now a language \mathcal{L} with a vocabulary V that contains terms to represent every concept in C . A logical model for \mathcal{L} can be defined as a structure $\langle S, I \rangle$: S is the structure $\langle D, R \rangle$, where D is the domain of individuals and R is a set of extensional relations; $I: V \rightarrow D \cup R$ is an interpretation function assigning elements of D to constant symbols in V , and elements of R to predicate symbols of V . A model, such as this one, fixes a particular extensional interpretation of language \mathcal{L} . Analogously, we can define an intensional interpretation by means of the

¹ We have so far used the term *model* instead of specification since it is the most common term in conceptual modeling. In this session, exclusively, we adopt the latter in order to avoid confusion with the term (logical) model as used in logics and Tarskian semantics. A specification here is a syntactic notion; a logical model is a semantic one.

structure $\langle C, \mathfrak{I} \rangle$, where $C = \langle W, D, \mathfrak{R} \rangle$ is a conceptualization and $\mathfrak{I}: V \rightarrow D \cup \mathfrak{R}$ is an intensional interpretation function which assigns elements of D to constant symbols in V , and elements of \mathfrak{R} to predicate symbols in V . In [8], this intensional structure is named the *ontological commitment* of language \mathcal{L} to a conceptualization C . We therefore consider this intensional relation as a formal characterization of the *represented by* relation depicted in figure 1, or simply a formal characterization of the Real-World Semantics of \mathcal{L} [10].

Given a logical language \mathcal{L} with vocabulary V , an *ontological commitment* $K = \langle C, \mathfrak{I} \rangle$, a model $\langle S, I \rangle$ of \mathcal{L} is said to be compatible with K if: (i) $S \in S_C$; (ii) for each constant c , $I(c) = \mathfrak{I}(c)$; (iii) there exists a world w such that for every predicate symbol p , I maps such a predicate to an admissible extension of $\mathfrak{I}(p)$, i.e., there is a conceptual relation ρ such that $\mathfrak{I}(p) = \rho$ and $\rho(w) = I(p)$. The set $I_K(\mathcal{L})$ of all models of \mathcal{L} that are compatible with K is named the set of *intended models* of \mathcal{L} according to K .

Finally, given a specification X in a specification language \mathcal{L} , we define as the logical rendering of X , the logical theory T that is the first-order logic description of that specification [11].

In order to exemplify these ideas let us take the example of a very simple conceptualization C such that $W = \{w, w'\}$, $D = \{\text{Gordon, Andy, Stewart}\}$ and $\mathfrak{R} = \{\text{person, father}\}$. Moreover, we have that $\text{person}(w) = \{\text{Gordon, Andy, Stewart}\}$, $\text{father}(w) = \{\text{Gordon}\}$, $\text{person}(w') = \{\text{Gordon, Andy, Stewart}\}$ and $\text{father}(w') = \{\text{Gordon, Stewart}\}$. This conceptualization accepts two possible state of affairs, which are represented by the world structures $S_w C = \{\{\text{Gordon, Andy, Stewart}\}, \{\{\text{Gordon, Andy, Stewart}\}, \{\text{Gordon}\}\}\}$ and $S_{w'} C = \{\{\text{Gordon, Andy, Stewart}\}, \{\{\text{Gordon, Andy, Stewart}\}, \{\text{Gordon, Stewart}\}\}\}$. Now, let us take a language \mathcal{L} whose vocabulary is comprised of the terms `Person` and `Father` with an underlying meta-model that poses no restrictions on the use of these primitives. In other words, the meta-model of \mathcal{L} has the following logical rendering (T_1): $\{\exists x \text{ Person}(x), \exists x \text{ Father}(x)\}$. In this case, we can clearly produce a logical model of \mathcal{L} (i.e., an interpretation that validates the logical rendering of \mathcal{L}) but that is not an intended world structure of C . For instance, the model $D' = \{\text{Gordon, Andy, Stewart}\}$, $\text{person} = \{\text{Gordon, Andy}\}$, $\text{father} = \{\text{Stewart}\}$, and $I(\text{Person}) = \text{person}$ and $I(\text{Father}) = \text{father}$. This means that we can produce a specification using \mathcal{L} which has a model that is not an *intended model* according to C .

Now, let us update the meta-model of language \mathcal{L} by adding one specific constraint and, hence, producing the meta-model (T_2): $\{\exists x \text{ Person}(x), \exists x \text{ Father}(x), \forall x \text{ Father}(x) \rightarrow \text{Person}(x)\}$. Contrary to \mathcal{L} , the resulting language \mathcal{L}' with the amended meta-model T_2 has the desirable property that all *its valid specifications have logical models that are intended world structures of C*.

A domain conceptualization C can be understood as describing the set of all possible state of affairs, which are considered admissible in a given universe of discourse U . Let V be a vocabulary whose terms directly correspond to the intensional relations in C . Now, let X be a *conceptual specification* (i.e., a concrete representation) of universe of discourse U in terms of the vocabulary V and

let T_X be a logical rendering of X , such that its formal constraints restricts the possible interpretations of the members of V . We call X (and T_X) an *Ontology* of U according to C iff the logical models of T_X describe all and only state of affairs which are admitted by C . This use of the term ontology is strongly related to a definition of Ontology put forth by the philosopher W.V.O. Quine, i.e. ontology as “*a theory concerning the kinds of entities and specifically the kinds of abstract entities that are to be admitted to a language system*” [2].

With an explicit representation of a conceptualization in terms of a suitable ontology, one can measure the truthfulness (or *domain appropriateness*) of a language \mathcal{L} to domain D , by observing the difference between the set of logical models of the (logical rendering of) meta-model M of \mathcal{L} and the set of logical models of the (logical rendering of) ontology O of D (see Figure 3). In the ideal case, these two specifications are isomorphic and, hence, share the same set of logical models. Therefore, not only every entity in conceptualization C must have a representation in the meta-model M of language \mathcal{L} , but these representations must obey the same axiomatization.

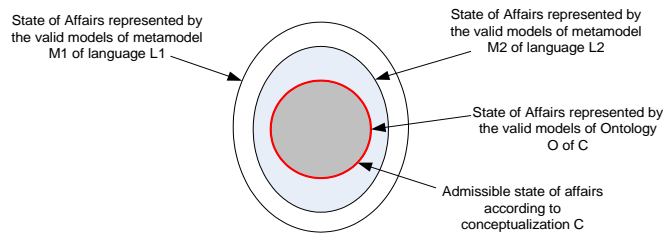


Fig. 3. Measuring the degree of *domain appropriateness* of modeling languages via an ontology of a conceptualization of that domain.

According to the language evaluation framework and the formal characterization of the relation between ontology and language vocabulary defined here, we can provide the following characterization for an ideal language to represent phenomena in a given domain according to a given reference ontology:

A language is ideal to represent phenomena in a given domain if the metamodel of this language is isomorphic to the reference ontology of that domain and the language only has as valid specifications those whose logical models are exactly the logical models of that reference ontology.

The traditional account of ontological analysis of languages in the literature are articulated in terms of isomorphism between language and ontology (such as in [5,6]). The above definition relates this traditional account with a formal definition of ontology as a *formal and explicit specification of a conceptualization* [8]. There is one direct manner in which incompleteness (and hence, lack of isomorphism) can impact the quality of language \mathcal{L} , namely, when the meta-model \mathcal{M} of \mathcal{L} does

not contain constructs to fully characterize a state of affairs, and therefore to produce the axiomatization necessary to exclude unintended logical models of the conceptualization at hand. To give one example, in the genealogical domain, without a gender differentiation for people, one cannot produce an axiomatization which excludes models in which people have two individuals of the same gender as their biological parents. Additionally, as exemplified in section 2, without the proper formal constraints in its meta-model, even lucid, sound, laconic and complete representation systems can be used to produce specifications lacking these desirable characteristics.

4 Successful Cases of General Conceptual Modeling Languages Evaluation and Re-Design using the Proposed Approach

The definition of an *ideal conceptual modeling language* given in section 3 provides clear guidelines for the design of the ontological meta-models of these languages. Given a reference ontology, the meta-model at hand should be isomorphic to the ontology of the domain. Moreover, it should include formal constraints such that the language would only accept as grammatically correct models those that represent state of affairs deemed admissible by the ontology at hand. Finally, this ontological meta-model should be further enriched with additional formal constraints that guarantee lucidity, laconicity, completeness and soundness for all individual diagrams that can be produced using that language.

These guidelines as advocated here are agnostic regarding the type of language which is being evaluated or (re)designed, meaning, this framework can be employed both for the case of general conceptual modeling languages (e.g., UML, ER, ORM, BPMN) and the case of *Domain-Specific Languages*. The type of language considered, however, is directly related to the type of ontology which can be used as a reference model. In the case of general (hence, domain-independent) conceptual modeling languages, the required reference ontology is a *Foundational Ontology*, i.e., a domain-independent system of categories and their ties which can be used to articulate models of different material domains in reality. In contrast, for the case of domain-specific languages, the required reference ontology is a *Domain Ontology* [2].

Two examples of Foundational Ontologies which have been successfully used for evaluating general conceptual modeling languages over the years are BWW [6,12] and UFO [10,13]: BWW has been used to analyze languages such as ARIS [14] and OWL [15], among others. The application of BWW for this purpose has traditionally been carried out by employing the original proposal put forth in [5]. However, none of these analyses have considered the axiomatization of the ontology (in terms of its admissible models) or the formal constraints incorporated in the language meta-models.

For a number of years, we have been analyzing conceptual modeling languages (including enterprise modeling languages), standards, environments and domain

ontologies, by employing the method described above and the foundational ontology UFO (Unified Foundational Ontology) as a reference model. The analyzed modeling languages include: UML [10, 16-19], Archimate [20], RM-ODP [21], TROPOS/i* and AORML [22,23], ARIS [24], and BPMN [25]. Despite the successful application of UFO in all these cases, it is important to highlight that the method discussed here could, in principle, be applied by taking different foundational ontologies as reference models. In fact, preliminary results on our ontological analysis of UML have been carried out with the foundational ontology GFO [26].

One significant case of ontological analysis using the framework discussed here and which deserves special attention is the case of UML. When considered as a Conceptual Modeling language, UML alone includes cases of all the anomalies discussed above. An example of *Construct Excess* in UML relates to the *Interface Construct*. As discussed in [10], being merely a design and implementation construct, there is no category in the reference ontology that serves as the ontological interpretation for a UML interface. Moreover, construct excess in the language level will cause unsoundness in all diagrams in which the exceeding construct is employed. UML also presents at least one case of *Non-Lucidity*, namely, in the *Association Class* construct. More than a case of *Construct Overload*, in each and every occasions this construct is used, it will stand for two ontological entities simultaneously, namely, a *Relator Universal* (e.g., Marriage, Enrollment, Employment) whose instances are individual relators (e.g., the Marriage of Mary and John, the Enrollment of Zoe to UFES) and a *Factual Universal* whose instance are tuples (e.g., pairs such as <John,Mary> and <Mary,John>) [10].

Another case of Construct Overload is the construct of *navigable ends* in UML which can be used to represent both *Relational Image Functions* (also known as mappings) and *Attribute Functions* [10]. Actually, also related to Attribute Functions, we have a case of *Construct Redundancy*, since attributes can be represented both by: the traditional textual representation of attributes spatially contained in the Class representation (figure 4.(a)) and navigable ends (figure 4.(b)).

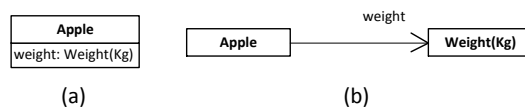


Fig. 4. Redundant representation of Attribute Functions in UML

Finally, cases of *Construct Deficit (Ontological Incompleteness)* in UML abound. For example, there are several different sorts of *object types* and *part-whole relations* in the conceptualizations proposed in [17] and [18], respectively, which are not directly represented by any construct of the language. In both cases, the distinct concepts present in the conceptualization are overloaded by the language constructs of *class* and *aggregation/composition*, respectively. To cite just one more example, in [19], we have shown that the concept of Mode (an ontological

counterpart to the ER notion of Weak Entity) also finds no direct representation in the language.

In [10], a philosophically and cognitively motivated foundational ontology (later identified as the type-fragment of UFO-A) has been used to re-design a complete version of the class diagrams fragment of the UML 2.0 meta-model giving rise to a well-founded version of UML for structural conceptual modeling and domain ontology representation. This ontology representation language (later dubbed *OntoUML*) has been successfully employed to create domain ontologies in several different industrial case studies in domains such as Telecommunications [27] and Energy (Petroleum and Gas) [28]. Moreover, it has been used to support meaning negotiation and semantic interoperability in the integration of ECG standards [29]. Furthermore, a version of this language has been employed over the past years by a department of the DoD in a significant number of successful applications in real-world engineering settings².

Asides from the discussed cases of Construct Overload, Excess, Deficit, Redundancy, and Non-Lucidity at the language level, the weakly constrained original UML meta-model accepts a number of instances which represent ontologically inadmissible ontological structures. All these problems have been addressed in [10] as well as in follow-up publications such as [30,33]. As a result, the revised UML meta-model (i.e., the *OntoUML* meta-model) is isomorphic to the ontological distinctions comprising the underlying foundational ontology and includes as formal constraints representations of the ontological constraints. Due to this strategy, these ontological distinctions and constraints could be directly implemented using meta-modeling architectures such as the OMG's MOF (Meta Object Facility)³. In this line, [31] reports on an implementation of an *OntoUML* graphical editor, which applies such an approach for assisting the user in creating ontologically correct models.

5 Ontological Meta-Properties, Model Simulation and Domain-Specific Visual Languages

Asides from being an extensive and successful evaluation case of the framework discussed here, *OntoUML* constitutes in itself a contribution to the application of this framework in the level of material domains and, thus, in the evaluation and design of *Domain-Specific Languages*. The most obvious reason is the following: the only grammatically correct models in *OntoUML* are ontologically consistent models; *OntoUML* can be used to represent structural conceptual models, in general, and domain ontologies in particular; thus, the domain ontologies constructed in *OntoUML* will be consistent with the axiomatization of the underlying foundational ontology.

²http://www.omgwiki.org/architecture-ecosystem/lib/exe/fetch.php?media=dmg_for_enterprise_ldm_v2_3.pdf

³<http://www.omg.org/mof/>

However, there are two other reasons for why this language plays an important role in the design of domain ontologies that will, in turn, be used for the evaluation and design of domain-specific visual modeling languages. Firstly, besides from its model-theoretical semantics defined in [10], OntoUML has an operational semantics defined as a mapping from this language to the lightweight formal language Alloy [32]. Due to this mapping, we have configured the Alloy Analyzer tool⁴ so that it can be used for supporting the modeler in assessing the gap between the *intended models* (in the sense of section 3) and the *possible models* of the domain ontology at hand (and, hence, of a possible meta-model isomorphic to it). This topic is discussed and illustrated in sections 5.2 and 5.3. Secondly, contrary to the merely formal (algebraic) structures employed in [3], the domain ontologies represented in OntoUML capture a number of subtle ontological meta-properties that are used to further qualify the ontological status of the domain concepts. In section 5.4, we illustrate how the ontological meta-properties can be systematically exploited to improve the system of concrete syntax of domain-specific visual modeling languages.

5.1 Ontological Meta-Properties

Due to space limitations, we concentrate here on a fragment of OntoUML, with a focus on distinctions among *Object Types* and *Part-Whole* relations spawned by variations in ontological meta-properties.

A fundamental modal meta-property used to distinguish among categories of *Object Types* is *Rigidity* (and the associated notion of *Anti-Rigidity*). Formally, we have that [17]: a type T is rigid iff every instance of T is necessarily an instance of T (in the modal sense). In contrast, a type T' is anti-rigid iff for every instance x of T' there is a possible situation in which x is not an instance of T' . In other words, an instance of a rigid type T cannot cease to instantiate it without ceasing to exist. Contrariwise, instances of T' only instantiate it contingently and, hence, can move in and out of the extension of T' without altering their identity. A stereotypical example that illustrates this distinction in most conceptualizations is marked by the types *Person* and *Student*: instances of *Person* are necessarily so (thus, *Person* is a rigid type); in opposition, instances of *Student* are merely contingently so (thus, *Student* is an anti-rigid type).

Object types that are rigid are named *Kinds* and *Subkinds* [17]. These types define a stable backbone, i.e., a taxonomy of rigid types instantiated by a given individual (the kind being the unique top-most rigid type instantiated by an individual).

Within the category of anti-rigid object types, we have a further distinction between *Phases* and *Roles* [17]. Both *Phases* and *Roles* are specializations of rigid types (*Kinds*/*subKinds*). However, they are differentiated w.r.t. their *specializa-*

⁴alloy.mit.edu/

tion conditions. For the case of Phases, the specialization condition is always an intrinsic one. For instance, a Child is a Person whose age is within a certain range. In contrast, the specialization condition for Roles is a relational one. For instance, a Student is a Person who is enrolled in an Educational Institution.

Again, a modal meta-property used to distinguish among the categories of Part-Whole relations is *Existential Dependence* [18]. We have that an entity x is existentially dependent on another entity y iff in every situation that x exists then y must exist. Associated to Existential Dependence we have the notion of Generic Dependence. We have that an entity x is generically dependent on a type Y iff in every situation where x exists an instance of Y must exist. These notions are used in UFO (among many other things) to distinguish between part-whole relations that imply existential dependence and those that only imply generic dependence. A part-whole relation which implies only generic dependence from the part to the whole is named *parthood with mandatory wholes* [18]. In contrast, a part-whole relation that implies existential dependence from the part to the whole is termed *inseparable parthood* [18]. A stereotypical example that illustrates this distinction in most conceptualizations is marked by the types of the relation between a Heart and a Person, on one side, and between a Brain and a *particular* Person, on the other: while a Heart needs to be part of an instance of Person (which does not have to be the same in every possible situation), a Brain needs to be part of a specific Person in all situations in which it exists.

Another remark regarding part-whole relations worth mentioning here is the following: contrary to purely formal mereological relations, part-whole relations which appear in conceptual models and material domain ontologies are non-transitive, i.e., they are transitive in certain situations and intransitive in others [33]. As illustrated in section 5.4, assessing the correct value of this additional meta-property of part-whole relations has an important influence in the design of their concrete visual representations.

Finally, part-whole relations can be distinguished according to a meta-property named *shareability*. This meta-property wrongly defined in the original UML specification has been refined in [10] with the following definition: (a) a (whole) type X is characterized by an exclusive (non-shareable) parthood relation with a (part) type Y iff every instance of X must have exactly one instance of Y as part; (b) a type X is characterized by a shareable parthood relation with a type Y iff instances of X can have more than one instance of Y as part.

5.2 Contrasting Possible and Intended Models with Visual Model Simulation

A modeling language such as OntoUML, incorporating the ontological constraints of a foundational theory, prevents the representation of ontologically non-admissible states of affair in domain ontologies represented in that language. However, it cannot guarantee that only *intended states of affairs* are represented

by the domain model at hand. This is because the admissibility of domain-specific states of affair is a matter of factual knowledge (regarding the world being the way it happens to be), not a matter of consistent possibility.

To illustrate this point, suppose a medical domain ontology representing the procedure of a transplant. In this case, we have domain concepts such as Person, Transplant Surgeon, Transplant, Transplanted Organ, Organ Donor, Organ Donee, etc. The (obviously incomplete) model of figure 5, which models aspects of this situation, does not violate any ontological rule. It would be the case, for example, had we placed Organ Donor as a super-type of Person, or represented the possibility of a Transplant without participants. These two cases can be easily detected and proscribed by an editor such as the one just proposed in [31]. However, there are still unintended states of affairs (according to a conceptualization assumed here) that are represented by valid instances of this model. One example is a state of affairs in which the Donor, the Donee and the Transplant Surgeon are one and the same Person. Please note that this state of affairs is only considered inadmissible due to domain-specific knowledge of social and natural laws. Consequently, it cannot be ruled out *a priori* by a domain independent system of ontological categories.

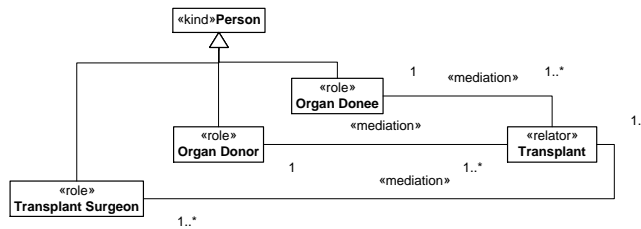


Fig.5. A fragment of a fictitious ontology in which unintended instances are admitted

Guaranteeing the exclusion of unintended states of affairs without a computational support is a practically impossible task for any relevant domain. In particular, given that many fundamental ontological distinctions are modal in nature, in order to validate a model, one would have to take into consideration the possible valid instances of that model in all possible worlds.

In [34], we have proposed an approach for OntoUML which offers a contribution to this problem by supporting conceptual model validation via visual simulation. On the one hand, it aims at proving the satisfiability of a given ontology by presenting a valid instance (logical model) of that ontology. On the other hand, it attempts to exhaustively generate instances of the ontology in a branching-time temporal structure, thus, serving as a visual simulator for the possible dynamics of entity creation, classification, association and destruction. The snapshots in this world structure confront a modeler with states of affairs that are deemed admissible by the ontology's current axiomatization. This enables modelers to detect unintended states of affairs and to take the proper measures to rectify the model. The assumption is that the example world structures support a modeler in this validation process, especially since it reveals how states of affairs change in time and how they may eventually evolve in counterfactual scenarios.

After running simulations of the model of figure 5, the model engineer would be presented with the consequences of her specification. When faced with a situation in which the Donor, Donee and Surgeon roles are played by the same person, she could realize that the ontology at hand has been *under-constrained* and then include a constraint in the model to exclude this unintended situation. Now, suppose the situation in which the modeler tries to rectify this model by declaring the types Transplant Surgeon, Organ Donor and Organ Donee as mutually disjoint. In a follow up execution of simulating this ontology, she would then realize that it is not possible, for example, for an Organ Donor to receive an organ in a different transplant, and for a Transplant Surgeon to be either an Organ Donor or an Organ Donee in different transplants. When facing this new simulation results, the modeler could realize that now the ontology has been *over-constrained*, after all there is no problem in having the same person as Organ Donor and Donee, or as Surgeon and Donor (Donee), they only cannot play more than one of these roles in the same transplant! In summary, the idea is that in this multi-step interaction with the model simulator, the modeler can keep refining the domain constraints to increasingly approximate the possible model instances of the ontology to those that represent admissible states of affairs according to the underlying conceptualization. In addition, in line with [35], we advocate that “*simulation helps catch errors of overconstraint, by reporting, contrary to the user’s intent, that no instance exists within the finite bounds of a given ‘scope’, or errors of underconstraint, ‘by showing instances that are acceptable to the specification but which violate an intended property’*”.

5.3 From a Domain Ontology to the Design of a Domain-Specific Conceptual Modeling Language Meta-model

In figure 6 below, we have a small ontology fragment in the domain of organizations represented using OntoUML. In the underlying conceptualization, *Employee* is a role played by people associated to one *Department*. People (instances of *Person*) are either instances of *Man* or of *Woman*, i.e., *Person* is an abstract type in the sense of object-oriented modeling, meaning there is no one who is a *Person* without being either a *Man* or a *Woman*. Every *Employee* is *part of* exactly one *Department* (represented by the non-shareable association end). However, since this is merely a generic dependence relation, employees can in principle change to different departments (even in different branches) in their lifecycle in the organization. An *Employee* is subordinated to at least one other employee who is his/her superior. In other words, the types *Subordinate Employee* and *Superior* are roles played by employees in the scope of hierarchical relations. As roles, these types are only contingently instantiated by their instances and the relational specialization condition here is represented by the *reports-to* relation. In other words, an instance of *Subordinate Employee* can cease to be one, and for her to instantiate this role, there must exist another *Employee* instantiating the *Superior Employee* role.

Moreover, the same instance of Employee can simultaneously instantiate both roles in the scope of different hierarchical relations (i.e., being the *Superior* of Employee A and subordinated to Employee B).

Every *Department* is *part of* exactly one *Organizational Branch*. Here, again, we have the case of a non-shareable parthood relation but also one which implies existential dependency from the part to the whole (represented by the *{inseparable}* tag value), i.e., the Sales Department of one Organizational Branch can only exist as part of that Branch. The relation between *Employee* and *Department*, on one hand, and *Department* and *Organizational Branch*, on the other, matches one of the cases of transitive parthood identified in [33]. For this reason, we have that: if an *Employee* E is part of a *Department* D and D is part of the *Organizational Branch* O then E is part of O.

Commissions are collectives that have particular *Employees* as members (accordingly termed *Commission Members*). Commissions can be in two different phases depending on the value of one of its intrinsic property (its amount of committed work). Thus, a *Work-Overloaded Commission* is a *Commission* such that its amount of committed work surpasses a certain threshold. Finally, a *Normal Workload Commission* is the complement of *Commission* w.r.t. *Work-Overloaded Commission*, i.e., its instances are all instances of the former which are not instances of the latter.

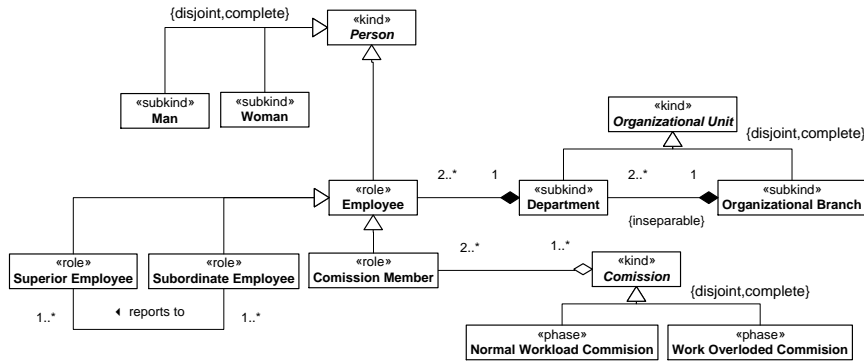
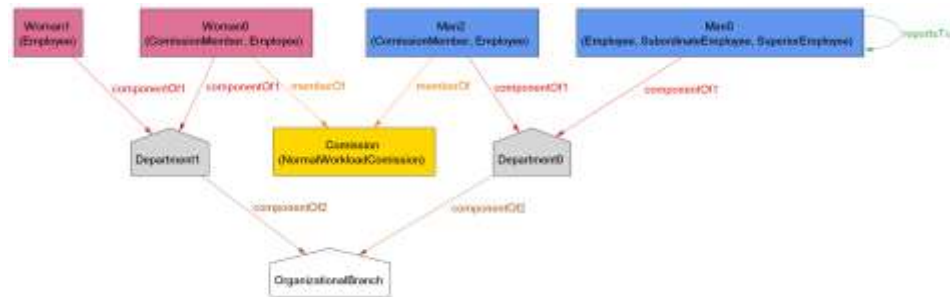


Fig.6 . A Domain Ontology for an Organizational Domain used to create a Meta-model for a Domain-Specific Language

Besides the representation of all relevant domain types, relations and properties of the underlying conceptualization, a domain ontology (and the meta-model derived from it) must include a body of formal constraints. This axiomatization must restrict the states of affairs represented by valid models of this ontology/meta-model to those which represent intended states of affairs according to the underlying conceptualization. As discussed in section 3, the quality of this ontology depends on the distance between these two sets of states of affairs. Moreover, as discussed in the previous section, for the case of OntoUML we can count on an approach for model validation via visual simulation.

Figures 7(a) and (b) present results of a visual simulation of the ontology in figure 6. By looking at these automatically generated models, the modeler can realize the lack of the missing partial order constraints that should be defined for the relation *reports to*. In figure 7(a), an Employee plays the roles of Superior Employee and Subordinate Employee in the same relation, i.e., the employee reports to herself. In figure 7(b), we can notice that Man0 is subordinate to both Woman2 and Woman0, who are then subordinate to Woman1, who then is subordinate to Man0. In other words, the model admits cycles in the *reports to* relation. Still on the model of figure 7(b), one can notice that although Man0 is subordinate to Woman2 and Woman0, both who are subordinate to Woman1. However, Man0 is not subordinate to Woman1, i.e., the *reports to* relation is not considered to be transitive. In figure 7(c), one can notice the possibility of an employee who falls outside the hierarchical structure of the organization, i.e., who is neither subordinate nor superior to anyone. This is due to a missing *{complete}* constraint in the generalization set from Employee to Superior Employee and Subordinate Employee. Finally, in the model of figure 7(d), one can notice the situation in which an employee (Man1) reports to a superior (Man0) of a different department. As previously discussed, these models cannot be deemed undesirable due to general ontological rules but only due to domain-specific rules. If we assume that in the conceptualization underlying the ontology of figure 6 these are all unintended models, then when facing them as possible ones, the modeler can improve the ontology (and corresponding Domain-Specific Language Meta-model) at hand by including the constraints required for their exclusion.



(a)



(b)

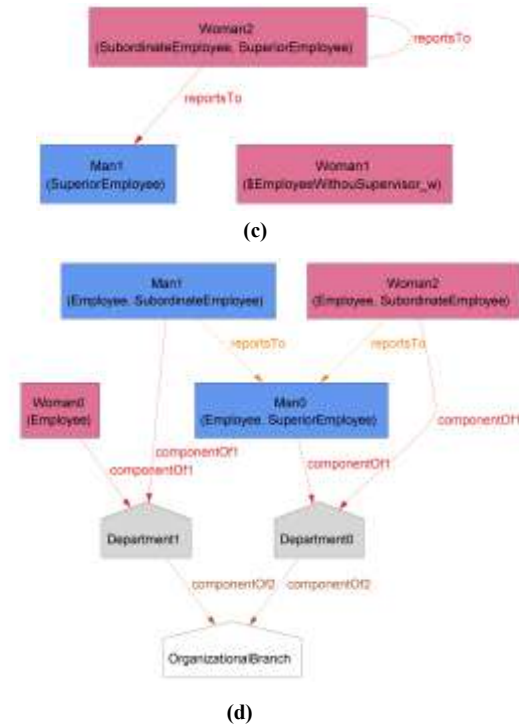


Fig.7 . Examples of possible but unintended instances of the Ontology in fig.6: (a) The employee that is his own superior; (b) cyclic hierarchies; (c) Employee outside the organizational structure; (d) Employee with a Superior in a different department.

5.4 From a Domain Ontology to the Design of Domain-Specific Visual Modeling Language Concrete Syntax

In this section, we discuss the impact that the reference ontology also has in the design of a concrete syntax for a visual conceptual modeling language. In order to do that, we base our discussion in the framework for analysis and design of Visual Languages put forth by Daniel Moody in [36,37].

The most direct influence that an ontology has on the visual notation of a language regards the quality that Moody terms *Semiotic Clarity*. By discussing Semiotic Clarity, Moody conducts an analysis similar to the one put forth here, but now relating ontology and visual concrete syntax. He draws from Nelson Goodman's *Theory of Symbols* when advocating "for a notation to satisfy the requirements of a notational system, there should be a one-to-one correspondence between symbols and their referent concepts" [38]. Figure 8 below (from [37]) summarizes this correspondence between the Ontological Meta-model of the lan-

guage and the underlying domain ontology, on one hand, and between the Ontological Meta-model of the language and the description of the concrete syntax, on the other.

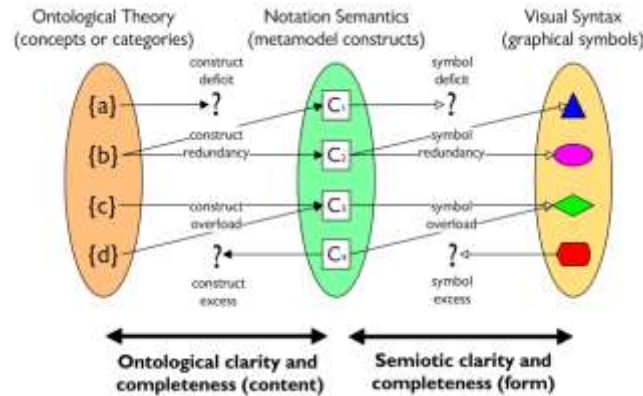


Fig.8. From Ontological Concepts to Language Primitives to Visual Syntax (from [37])

In classifying the anomalies that take place when the isomorphism between the latter pair of models is broken, Moody builds explicitly on the vocabulary used in the literature of ontological analysis: (a) *Symbol redundancy* exists when multiple symbols are used to represent the same semantic construct; (b) *Symbol overload* exists when the same graphical symbol is used to represent different semantic constructs; (c) *Symbol excess* exists when graphical symbols are used that do not represent any semantic construct; (d) *Symbol deficit* exists when semantic constructs are not represented by any graphical symbol.

The problems caused by these anomalies are, as explained by Moody, also analogous to those founded when the isomorphism between ontology and abstract syntax is broken: symbol redundancy places a burden of choice on the language user to decide which symbol to use and an additional load on the reader to remember multiple representations of the same construct; Symbol overload leads to ambiguity and the potential for misinterpretation [38]; symbol excess unnecessarily increases graphic complexity, which has been found to reduce understanding of notations [39]. Moreover, if symbol deficit exists, the visual notation is said to be *semiotically incomplete*. If any of the other three anomalies exist, the notation is *semiotically unclear*.

There is an obvious connection here with what we have been discussing so far: the suitability of a visual notation is evaluated w.r.t. a system of modeling primitives, which in turn is evaluated w.r.t. to a domain ontology. Hence, the quality of a system of visual syntax w.r.t. semiotic clarity indirectly but essentially depends on the characteristics of the underlying domain represented in that domain ontology. One aspect, however, which is not evident in Moody's model above, is the following. As previously discussed, the graphical symbols which form the system of concrete syntax often fall naturally into a hierarchical typing which informs about the semantics of what is being represented. An analogous statement can be made

regarding certain relations between graphical symbols (e.g., spatial relations) that can be systematically mapped onto semantic relations with equivalent logical properties. This feature of graphical symbol systems and relations is illustrated by the examples in the sequel.

We start with a first example illustrated in figure 9. As one can notice, the models of figure 9.(a) and 9.(b) are isomorphic. The different concrete kinds of entities in the model (Federal Capital, State and City) are represented by different kinds of geometrical objects (Non-Squared Rectangle, Square and Circle). In particular, the taxonomic structure of Geopolitical Units is isomorphic to the one of Geometric Figures. For this reason, in a visual query one can immediately notice that Federal Capital is more similar to a State than to a City and probably share a common super-type with the former. Notice that, had we produced a different taxonomic structure on the model of figure 9.(a), then a different choice of representing graphical symbols would have been made possibly creating undesired implicatures for the model reader. Given the difficulties experienced by modelers in the design of domain taxonomic structures [17], this illustrates the importance of having a well-designed ontology for the design of semiotic clarity in the system of visual syntax.

There is another point worth mentioning about this example. As discussed in the previous section, a phase represents a type that is instantiated by an individual only contingently (in the modal sense) and changes of phases are motivated by changes in intrinsic properties. In this example, the same city can be considered a town in a world w and a metropolis in w' while still maintaining its cross-world identity. Likewise, in Figure 9.(b), the size property of a geometric figure is considered one of its contingent intrinsic properties. Thus, a particular circular form is assumed to be able to change its size while maintaining a continuous visual percept. Furthermore, the intrinsic property *population size* that motivates the phase changes of cities is associated with a linearly ordered dimension. For this reason, we have decided to associate the intrinsic property of Circles that represent this phase variation by employing also a linearly ordered dimension (size).

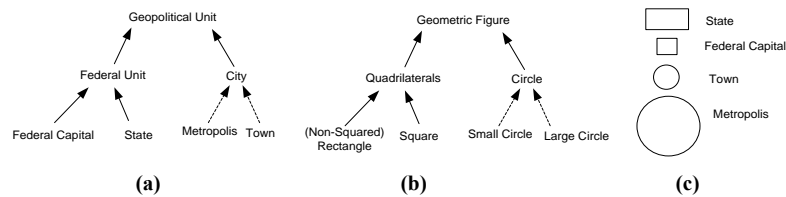


Fig.9. (a) a fragment of taxonomy for a geopolitical domain; (b) a taxonomy of geometric objects isomorphic to the structure in (a); (c) a system of visual symbols from (b) to represent the domain concepts in (a)

As a second example, we refer to the model of figure 6. In figure 10 below, we present a model in a domain-specific visual language designed to represent valid instances of the ontology of figure 6. There are a number of aspects in the concrete syntax of this visual language that have been designed by systematically considering logical and ontological meta-properties of the domain concepts in figure 6.

Firstly, this system of concrete syntax possesses Semiotic Clarity, i.e., there is a one-to-one correspondence between its categories and the domain types and relations represented in figure 6. Secondly, the mapping between the domain elements and the elements in the visual notation takes full account of the ontological categories and meta-properties of the former. In the sequel, we elaborate on this second point:

Kinds and Subkinds: In the model of figure 6, we have three kinds of elements, namely, Person, Organizational Units and Commission. Since Person is an abstract type, we have that all instances of this type in the domain are instances of either Man or Woman. Likewise, all instances of Organizational Units are either Organizational Branches or Departments. In summary, all the concrete substantial entities in this domain are either instances of Man, Woman, Departments, Organizational Branches or Commissions.

As discussed in [40], shapes defined by closed contour are among the most basic metaphorical representations for objects. This idea is in line with a number of findings in cognitive science, including the one that shape plays a fundamental role in kind classification [41] (infants will tend to classify things as being of the same kind if they share a similar shape). Moreover, our most primitive notion of object (in fact, our most primitive sortal type) is the notion of a maximally-topologically-self-connected object which moves in a spatial temporal trajectory together with all its parts [42]. This idea is directly represented by convex shapes with closed boundaries.

In the language used in figure 10, each distinct concrete (sub)kind of domain objects is associated with a shape in the aforementioned sense. Moreover, the chosen shapes are sufficiently dissimilar and are aligned with the taxonomic relations between domain types as presented in figure 10. For example, the “four-sided” figures used to represent Organizational Branches and Departments are similar, respecting the fact that they are both direct subtypes of organizational units. On the other hand, they are dissimilar from the blobs used to represent Commissions and the Icons used to represent People. These features allow for another important quality characteristic discussed by Moody [36], namely, *Perceptual Discriminability*. For this reason, by looking at a model in this language one can immediately tell which domain element is being represented by which graphical element.

A final aspect worth mentioning is the direct metaphorical resemblance between the graphical elements used and their referents. The most obvious case is the iconic representation for Man and Woman. However, the representation of Departments as “pieces of an Organizational Branch” is adherent to the idea of “Organizational Divisions” associated to Departments, Sectors, etc. In addition, while the straight lines used in the contour of Organizational Units gives the idea of more formal and rigid structure, the round boundaries of the blob representing Commissions is more naturally associated with a flexible informal one. The systematic use of these metaphorical resemblances brings to this notational system another important quality characteristic according to Moody, namely, *Perceptual Immediacy* [36]. In this example, by looking at the icons used to represent Man and Woman, one can directly infer the type of referent which is being represented.

Phases: As previously discussed, phases represent contingent specializations of kinds such that the specialization condition is related to the changes of value in the intrinsic properties of the instances of that kind. Accordingly, here once more we use an intrinsic property of visual percept used to represent the kind to represent different phases associated with that kind, i.e., the entity can be seen as changing phases but maintaining its identity due to the persistence of the visual percept. In the example of the language used in figure 10 the changes in color of the Blob used to represent Commissions represent different phases of a Commission. Moreover, we use a high-saturation color to represent the *Work-Overloaded Commission* exploring a metaphorical relation between “more quantity of color” (which is the definition of saturation) and “more quantity of work”. Once more, this feature of the graphical grammar increases its *Perceptual Immediacy*. Moreover, the difference in brightness of the grey hue used to represent an overloaded commission, on one side, and the white one used to represent a regular load commission on the other creates an efficient *Perceptual Pop-Out* [36], decreasing the cognitive cost of identifying former types of commissions in visual queries [40]. Finally, given that identifying overload commissions is an important task in this domain, the *perceptual pop-out* at hand is increased by the increased *perceptual discriminability* between these two phases. This is due to the use of a different in thickness of the blob boundaries. This is a case of what is termed *Dual Coding* in Moody’s work [36] and it constitutes a small deviation of Semiotic Clarity for the specific purpose of increasing efficiency in particularly important visual queries.

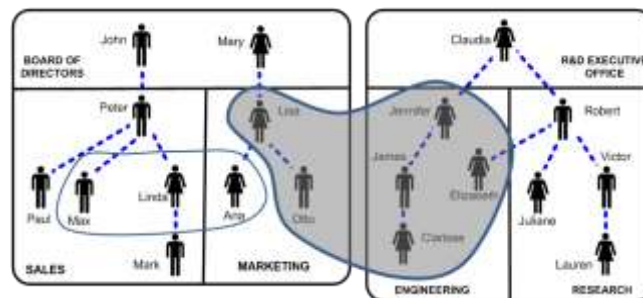


Fig.10 . A model in the a Domain-Specific Language to represent Organizational Structures

Relations: In the ontology of figure 6, we have three types of relations. Firstly, we have the relations of *component-of* parthood between (1) Employee and Department, and (2) Department and Organizational Branch. As discussed in [33], *componentOf* is irreflexive and asymmetric. Moreover, transitivity holds across (1) and (2). By using the relation of spatial inclusion in the plane to represent these relations, we have a mapping to a visual relation that has exactly the same formal properties of the represented one, since spatial inclusion is also a partial order relation. This feature of the visual notation allows for a direct *inferential free ride* [43]: when identifying some as being *part of* the Marketing Department in Organ-

izational Branch A, we immediately identify this person as being *part of* Organizational Branch A.

A second aspect that we would like to point out is that the different departments that comprise an Organizational Branch are represented by a tessellation of the spatial region used to represent that Branch. The lack of overlap between these regions allows for a *perceptually immediate* representation of the non-shareability meta-property of these *component-of* relations. In other words, since Departments are represented by tessellations (with non-overlapping regions), it becomes perceptually immediate to the user of the language that Employees are part of at most one Department and that Departments are parts of at most one Organizational Branch. This representation also contributes to *perceptual immediacy* due to yet another reason, namely, that if Departments are represented as *partitions of the region* representing its associated Branches, this also favors the interpretation of existential dependence (inseparability) from the part to the whole. To put it in a different way, it is much easier to visualize the icon for Man and Woman moving in and out of the Branch region than to visualize a piece of the region moving to another region.

Another type of parthood relation used in figure 6 is the one between Commission Member and Commission. Once more, this relation is also represented as a spatial containment relation between the icons representing People and the blob representing the Commission. However, as one can notice in figure 10, these blob forms can overlap with Branch and Department regions. This feature allows for the direct inferential free ride on the identification of which departments and branches commission members belong to. In addition, in line with the *shareability* meta-property of this relation in the ontology, one can easily imagine overlapping blobs allowing for a certain member to be simultaneously part of multiple commissions.

A third relation in this ontology is the *reports to* relation, defined between a (superior) employee and its subordinates. As previously discussed, this relation also defines a partial order relation between Employees. Here, we used a combination of visual relations to represent this domain association, namely, we combined the *above* relation in the plane (which is a total order relation) with the transitive closure of the *is-dashed-line-connected* relation. The combined relation is also a partial order relation. Additionally, the different texture of this line increases the *Perceptual Discriminability* when contrasting it to the solid lines used to demarcate department partitions. Finally, the spatial metaphor of using “higher in the plane” to represent “higher in the hierarchy” favors *Perceptual Immediacy* in this representation.

Roles and Relational properties: roles represent contingent specializations of kinds which, in contrast with phases, have a relational specialization condition, i.e., a role R_1 is played by entities of type A when associated via a certain relation T to entities of type B, typically playing a role R_2 [17]. Accordingly, roles R_1 and R_2 should be represented by a visual relation between the visual representations of A and T, and B and T, respectively. This strategy has been employed consistently for the representation of all roles in this visual notation. For example, the role of

Employee is represented by the *contained in region* relation between a People icon and the region representing the Branches. *Mutatis Mutandis*, the same can be said for the role of Commission Member. Finally, the complementary roles of *Superior* and *Subordinate* are accordingly represented by the adjacency relation between the People icons and the terminations of the dashed line representing the *reports-to* relation. The otherwise symmetric feature of this line (possibly suggesting a symmetric relation) is broken by the above relations between the icons in the plane. This asymmetry could be highlighted by the use of an asymmetric connector line. However, since asymmetry is already guaranteed by the combined relation, and our visual cognition is particular efficient to spot vertical misalignment, we advise against such a design choice. Especially since this choice not only would hurt *Semiotic Clarity* but also what Moody terms *Graph Parsimony*. This is captured in the following quote from [36]: “*Empirical studies show that increasing graphic complexity significantly reduces understanding of software engineering diagrams by naïve users...It is also a major barrier to learning and use of a notation*”.

In table 1 below, we present a correspondence between the concrete elements in the ontology of figure 6 and the system of graphical symbols comprising a visual modeling language used in figure 10.







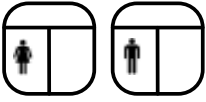



6 Final Considerations

In this chapter, we elaborate on the relation between a modeling language and a set of real-world phenomena that this language is supposed to represent. We focus on two aspects of this relation, namely, the *domain appropriateness*, i.e., the suitability of a language to model phenomena in a given domain, and its *comprehensibility appropriateness*, i.e., how easy it is for a user of the language to recognize what that language’s constructs mean in terms of domain concepts and how easy it is to understand, communicate and reason with the specifications produced in that language. We defend that both of these properties can be systematically evaluated for a modeling language w.r.t. a given domain in reality by comparing a concrete representation of the worldview underlying this language (captured in a meta-model of the language), with an explicit and formal representation of a conceptualization of that domain, or a *reference ontology*.

We therefore present a framework for language evaluation and (re)design which aims, in a methodological way, to approximate or to increase the level of homomorphism between a meta-model of a language and a reference ontology. This framework comprises a number of properties (*lucidity, soundness, laconicity, completeness*) that must be reinforced for an isomorphism to take place between these two entities. The framework proposed combines two existing proposals in the literature: (i) the one presented in [3,4], which focuses on the evaluation of individual representations and (ii) the one of [5,6], which aims at the evaluation of representation systems. In addition, our framework extends these two proposals in

several ways and, compared to them, our framework possesses a number of advantages discussed in the sequel.

Table 12. Visual Concrete Syntax for the Organization Structure Ontology of Fig.12

Domain Type	Ontological Category	Notational Element
Person	Kind	Abstract Class; No direct representation
Man	Subkind	
Woman	Subkind	
Organizational Unit	Kind	Abstract Class; No direct representation
Organizational Branch	Subkind	
Department and Department is component of Organizational Branch	subkind and part-whole relation	
Comission	Kind	Abstract Class; No direct representation
Normal Load Comission	Phase	
Overloaded Comission	Phase	
Employee and Employee role and part is component of Department	whole relation	
Superior	Role	
Subordinate Employee	Role	
Comission Member and Comission Member is part of Comission	role and part-whole relation	
SubordinateEmployee reports to Superior	Domain association	combination of is-dashed-line-connected with the above relation in the plane

The approach of Gurr uses regular algebraic structures to model a domain conceptualization. We strongly defend the idea that the more we know about a domain the better we can evaluate and (re)design a language for domain and comprehensi-

bility appropriateness. As we show in this chapter, there are important meta-properties of domain entities (e.g., rigidity, relational dependency) that are not captured by ontologically-neutral mathematical languages (such as algebras or standard set-theories), and that the failure to consider these meta-properties hinders the possibility of accounting for important aspects in the design of efficient visual pragmatics for visual modeling languages. By demonstrating how these ontological meta-properties could be used for these purposes, this chapter also contributes towards the construction of a systematic connection with the framework for the evaluation and design of concrete visual syntaxes proposed by Moody in [36,37].

The approach of Wand and Weber focuses solely on the design of general conceptual modeling languages. The framework and the principles proposed here instead can be applied to the design of conceptual modeling languages irrespective to which generalization level they belong, i.e., it can be applied both at the level of material domains and corresponding domain-specific modeling languages, and at the (meta) level of a domain-independent (meta) conceptualization that underpins a general conceptual (ontology) modeling language. For the case of domain-independent meta-conceptualizations, the framework discussed here has been applied to a number of prominent approaches (e.g., UML, Archimate, Tropos/i*, AORML, BPMN, ARIS, RM-ODP) as mentioned in section 4 of this chapter. For the case of domain-specific conceptualizations, this ontology-based framework amounts to an important contribution to the area of domain-specific languages design methodologies (as acknowledged, for instance, in [44,45,46]).

As in the original proposal of Wand and Weber, the focus of our framework is on the level of systems of representations, i.e., on the evaluation of modeling languages, as opposed to a focus on individual diagrams produced using a language. Nevertheless, as it is demonstrated here, by considering desirable properties of the mapping of individual diagrams onto what they represent, we are able to account for desirable properties of the modeling languages used to produce these diagrams, extending in this aspect Wand and Weber's work.

Finally, both the approaches of Gurr and Wand and Weber address solely the relation between ontological categories and the modeling primitives of a language, paying no explicit attention to the possible constraints governing the relation between these categories. Moreover, they do not consider the necessary mapping from these constraints to equivalent ones, to be established between the language constructs representing these ontological categories. As demonstrated in section 3 of this chapter, the proposal presented here, in contrast, explicitly considers the constraints governing the relations between the elements comprising a given domain conceptualization, and how these constraints are taken into account in a representation system. In particular, we demonstrate here how a strategy based on visual simulation can be used to validate this aspect of reference ontologies and the language meta-models based on them.

In summary, as a contribution for Domain Engineering, we presented a framework that can be used for the evaluation and (re)design of reference models, in general, and *Domain Models*, in particular. This approach can be employed to formally and systematically improve the quality of domain models that, in turn,

can be used to derive the meta-models of Domain-Specific Conceptual Modeling Languages. However, the use of an ontology-based method for producing high-quality domain models which encompass consistent, comprehensive and truthful domain axiomatizations is expected to have a significant impact in other domain engineering enterprises such as the principled design of domain frameworks [47].

Acknowledgments The author is grateful to Renata S.S. Guizzardi, João Paulo Almeida, Tiago Prince, Alessandro Benevides, Bernardo Braga, Luis Ferreira Pires, Marten van Sinderen and Nicola Guarino for the different lines of collaboration which are reflected in the topics of this chapter. In particular, he would like to express his gratitude to Gerd Wagner for the great number of insightful inputs over 10 years of fruitful collaboration. This research is supported by FAPES (PRONEX #52272362/2010) and CNPq (productivity grant #311578/2011-0).

References

1. Guizzardi, G., Ferreira Pires, L., van Sinderen, M.: An Ontology-Based Approach for Evaluating the Domain Appropriateness and Comprehensibility Appropriateness of Modeling Languages, ACM/IEEE 8th International Conference on Model Driven Engineering, Languages and Systems, LNCS, vol. 3713, Springer, 2005.
2. Guizzardi, G.: On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models, *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*, Olegas Vasilecas, Johan Edler, Albertas Caplinskas (Editors), ISBN 978-1-58603-640-8, IOS Press, Amsterdam, 2007.
3. Gurr, C.A.: Effective Diagrammatic Communication: Syntactic, Semantic and Pragmatic Issues., *Journal of Visual Languages and Computing*, 10, 317-342, 1999.
4. Gurr, C.A.: On the isomorphism, or lack of it, of representations, In: *Theory of Visual Languages*, Chapter 10 (K. Marriot & B. Meyer, eds), Springer, Berlin, 1998.
5. Wand, Y. and Weber, R.: An ontological evaluation of systems analysis and design methods, In E. Falkenberg and P. Lingreen (ed.), *Information System Concepts: An In-Depth Analysis*. Elsevier Science Publishers B.V., North-Holland, 1989.
6. Weber, R.: *Ontological Foundations of Information Systems*. Coopers & Lybrand, Melbourne, 1997
7. Grice, H.P.: Logic and conversation. In: *Syntax and Semantics: Vol 3, Speech Acts* (P. Cole & J. Morgan, eds). Academic Press, New York, pp. 43-58, 1975.
8. Guarino, N.: *Formal Ontology in Information Systems*, *Formal Ontology in Information Systems. Proceedings (FOIS)*, Italy, IOS Press, 1998.
9. Fitting, M., Mendelsohn, R. L.: *First-Order Modal Logic*, Kluwer Academic Publishers, Norwell, 1999.
10. Guizzardi, G.: *Ontological Foundations for Structural Conceptual Models*, Universal Press, The Netherlands, 2005. ISBN 1381-3617.
11. Ciocoiu, M., Nau D.: *Ontology-Based Semantics*. 7 th International Conference on Principles of Knowledge Representation and Reasoning (KR'2000), USA, 2000.
12. Wand, Y. & Weber, R.: Mario Bunge's Ontology as a formal foundation for information systems concepts, In Weingartner, P. & Dorn, G.J.W. (eds.), *Studies on Mario Bunge's Treatise*, Rodopi, Atlanta, 1990.

13. Guizzardi, G., et al: Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The Case of the ODE Software Process Ontology. In 11th Ibero-American Conference on Software Engineering (CIBSE), Recife, 2008.
14. P. Green, M. Rosemann: Integrated process modeling: An ontological evaluation, *Information Systems*, vol. 25 (2), 2000, pp. 73-87.
15. Bera, P., Wand, Y.: Analyzing OWL using a Philosophy-Based Ontology, 3rd International Conference on Formal Ontology in Information Systems (FOIS 2004), Torino, 2004.
16. Guizzardi, G., Wagner, G.: What's in a Relationship: An Ontological Analysis, 27th International Conference on Conceptual Modeling (ER'2008), Barcelona, Spain, *Lecture Notes in Computer Science*, v.5231, p.83 - 97, 2008.
17. Guizzardi, G.; Wagner, G.; Guarino, N.; van Sinderen, M.: An Ontologically Well-Founded Profile for UML Conceptual Models, 16th International Conference on Advances in Information Systems Engineering (CAiSE), Latvia, LNCS 3084, Springer-Verlag, 2004.
18. Guizzardi, G.: Modal Aspects of Object Types and Part-Whole Relations and the de re/de dicto distinction, 19th International Conference on Advanced Information Systems Engineering (CAISE'07), Trondheim, 2007, LNCS 4495, Springer-Verlag.
19. Guizzardi, G.; Masolo, C.; Borgo, S.: n the Defense of a Trope-Based Ontology for Conceptual Modeling: An Example with the Foundations of Attributes, Weak Entities and Datatypes, 25th International Conference on Conceptual Modeling (ER'2006), Arizona, USA, 2006.
20. Azevedo, C.; Almeida, J.P.A., van Sinderen, M.J., Quartel, D., Guizzardi, G.: An Ontology-Based Semantics for the Motivation Extension to ArchiMate, 15th IEEE International Conference on Enterprise Computing (EDOC 2011), Helsinki, Finland.
21. Almeida, J.P.A., Guizzardi, G.: An Ontological Analysis of the Notion of Community in the RM-ODP Enterprise Language Original Research Article, *Computer Standards & Interfaces*, Elsevier, 2012. doi: 10.1016/j.csi.2012.01.007.
22. Guizzardi, R.S.S., Guizzardi, G.: Ontology-Based Transformation Framework from Tropos to AORML, In: P. Giorgini; N. Maiden; J. Mylopoulos; E. Yu. (Org.), *Social Modeling for Requirements Engineering*, Cooperative Information Systems Series. Boston: MIT Press, 2010.
23. Guizzardi, R.S.S., Franch, X., Guizzardi, G.: Applying a Foundational Ontology to Analyze the i* Framework, 6th International Conference on Research Challenges in Information Systems (RCIS 2012), Valencia.
24. Santos Jr., P.S.S., Almeida, J.P.A., Guizzardi, G.: An Ontology-Based Semantic Foundation for ARIS EPCs, Proc. 25th ACM Symp. on Applied Computing (Enterprise Engineering Track), 2011.
25. Guizzardi, G., Wagner, G.: Can BPMN be used for Simulation Models?, 7th International Workshop on Enterprise & Organizational Modeling and Simulation (EOMAS 2011), together with the 23rd International Conference on Advanced Information System Engineering (CAiSE'11), London, UK.
26. Guizzardi, G., Herre, H., Wagner G.: On the General Ontological Foundations of Conceptual Modeling, 21st International Conference on Conceptual Modeling (ER), Finland, LNCS 2503, Springer-Verlag, 2002.
27. Barcelos, P.P.F.; Guizzardi, G.; Garcia, A.S., Monteiro, M.E.: Ontological Evaluation of the ITU-T Recommendation G.805, 18th International Conference on Telecommunications (ICT 2011), IEEE Press, Cyprus, 2011.
28. Guizzardi, G.; Lopes, M.; Baião, F.; Falbo, R.: On the importance of truly ontological representation languages, *International Journal of Information Systems Modeling and Design (IJISMD)*, 2010. Vol.1, no. 2, pp. 1-22, ISSN: 1947-8186.

29. Gonçalves, B. ; Guizzardi, G. ; Pereira Filho, J. G.: Using an ECG reference ontology for semantic interoperability of ECG data. *Journal of Biomedical Informatics*, v. 43, 2010.
30. Costal, D., Gomez, C., Guizzardi, G.: Formal Semantics and Ontological Analysis for Understanding Subsetting, Specialization and Redefinition of Associations in UML, 30th International Conference on Conceptual Modeling (ER 2011), Brussels, 2011.
31. Benevides, A. B., Guizzardi, G.: A Model-based Tool for Conceptual Modeling and Domain Ontology Engineering in OntoUML, 11th International Conference on Enterprise Information Systems (ICEIS), Springer, Milan, 2009.
32. Jackson, D.: *Software Abstractions: Logic, Language, and Analysis*, MIT Press, 2006.
33. Guizzardi, G.: The Problem of Transitivity of Part-Whole Relations in Conceptual Modeling Revisited, Proc. 21st International Conference on Advanced Information Systems Engineering (CAISE'09), Amsterdam, The Netherlands, 2009.
34. Benevides, A.B.; Guizzardi, G.; Braga, B.F.B.; Almeida, J.P.A.: Validating modal aspects of OntoUML conceptual models using automatically generated visual world structures, *Journal of Universal Computer Science*, vol. 16, no. 20 (2010), pp. 2904-2933.
35. Jackson, D.: Alloy: a Lightweight Object Modelling Notation, TOSEM (Transactions on Software Engineering and Methodology), ACM, New York, 11, 2 (2002), 256-290.
36. Moody, D.L., van Hillegerberg, J., Evaluating the Visual Syntax of UML: An Analysis of the Cognitive Effectiveness of the UML Family of Diagrams. 1st International Conference on Software Language Engineering (SLE), 2008: 16-34
37. Moody, D.L., The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering*. 35(6): 756-779 (2009)
38. Goodman, N. *Languages of Art: An Approach to a Theory of Symbols*. Bobbs-Merrill Co, Indianapolis, 1968.
39. Nordbotten, J.C. and Crosby, M.E.: The Effect of Graphic Style on Data Model Interpretation. *Information Systems Journal*, 9 (2). 139-156, 1999.
40. Ware, C.: *Visual Thinking for Design*, Morgan Kaufmann, 2008.
41. Tversky, B., Hemenway, K.: Objects, parts, and categories.. *Journal of Experimental Psychology: General*, 113, 169-193, 1984.
42. Xu, F., Carey, S.: "Infants' Metaphysics: The Case of Numerical Identity"; *Cognitive Psychology*, 30 , 5 (1996).
43. Shimojima, A.: Operational constraints in diagrammatic reasoning, In: *Logical Reasoning with Diagrams*. Oxford University Press, New York, pp. 27-48, 1996.
44. Becker,J.; Pfeiffer,D.; Räckers, M.: PICTURE – A new Approach for Domain-Specific Process Modelling, Proceedings of the CAISE'07 Forum at the 19th International Conf. on Advanced Information Systems Engineering Trondheim, Norway, 2007.
45. McDermott, J., Allwein, G.: A formalism for visual security protocol modeling, *Journal of Visual Languages & Computing*, Volume 19, Issue 2, April 2008, Pages 153–181.
46. McDermott, J.: Visual Security Protocol Modeling, Proceedings of the 2005 workshop on New security paradigms, New York, 2005.
47. Falbo, F., Guizzardi G., Duarte, K.: An Ontological Approach to Domain Engineering, Proceedings of 14th International Conference on Software and Knowledge Engineering (SEKE), ACM, New York, 2002.