An Ontology-Based Approach for Evaluating the *Domain Appropriateness* and *Comprehensibility Appropriateness* of Modeling Languages

Giancarlo Guizzardi, Luís Ferreira Pires, Marten van Sinderen

Centre for Telematics and Information Technology,
University of Twente, Enschede, the Netherlands
{guizzard, pires}@cs.utwente.nl, sinderen@ctit.utwente.nl

Abstract. In this paper we present a framework for the evaluation and (re)design of modeling languages. We focus here on the evaluation of the suitability of a language to model a set or real-world phenomena in a given domain. In our approach, this property can be systematically evaluated by comparing the level of homomorphism between a concrete representation of the worldview underlying the language (captured in a metamodel of the language), with an explicit and formal representation of a conceptualization of that domain (a reference ontology). The framework proposed comprises a number of properties that must be reinforced for an isomorphism to take place between these two entities. In order to illustrate the approach proposed, we evaluate and extend a fragment of the UML static metamodel for the purpose of conceptual modeling, by comparing it with an excerpt of a philosophically and cognitive well-founded reference ontology.

1 Introduction

The objective of this paper is to discuss the design and evaluation of artificial 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 w.r.t. a given real-world domain [1]: (i) *domain appropriateness*, which refers to truthfulness of the language to the domain; (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 *domain 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.

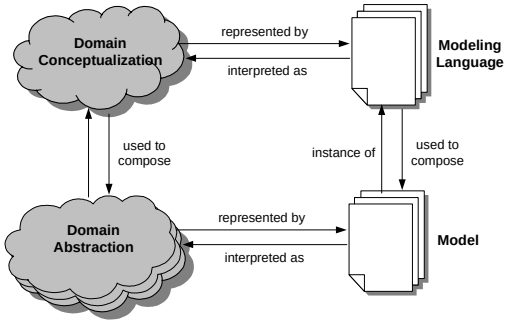

Figure 1. Relation between conceptualization, abstraction, modeling language and model.

In this paper, we propose a framework to evaluate the suitability of a language to model a set or 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 *metamodel of the language*), with an explicit and formal representation of a conceptualization of that domain (a *reference ontology* [8]). Our framework comprises a number of properties that must be reinforced for an isomorphism to take place between these two entities. If 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 [2], which focus on evaluating the match between individual diagrams and the state of affairs they represent, and the approach of [3], 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 [3]. We also build here on the work of the philosopher of language H.P.Grice [4] and his notion of *conversational maxims* that states that a speaker is assumed to make contributions in a dialogue which are *relevant*, *clear*, *unambiguous*, and *brief*, *not overly informative* and *true according to the speaker's knowledge*. Finally, in comparison to [2] and [3], 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 remaining of this paper is structured as follows. Section 2 introduces the evaluation framework proposed here. Section 3 presents a formal characterization of the notions of domain conceptualization and their representing ontologies, as well as their relations to modeling languages and particular models. In order to illustrate our approach, we evaluate and extend a fragment of the UML static metamodel for the purpose of conceptual modeling, by comparing it with an excerpt of a philosophically

and cognitive well-founded reference ontology. Section 4 discusses the foundational ontology employed for this purpose. Section 5 discusses the evaluation of the UML metamodel, and the extensions that we propose in order to enforce suitability to conceptual modeling. Section 6 presents some final considerations.

2 A Framework for Language Evaluation

Following [2], we define four properties that should hold for an isomorphic correlation to take place: *lucidity*, *soundness*, *laconicity* and *completeness* (see Figure 2). Each of these properties is discussed below.

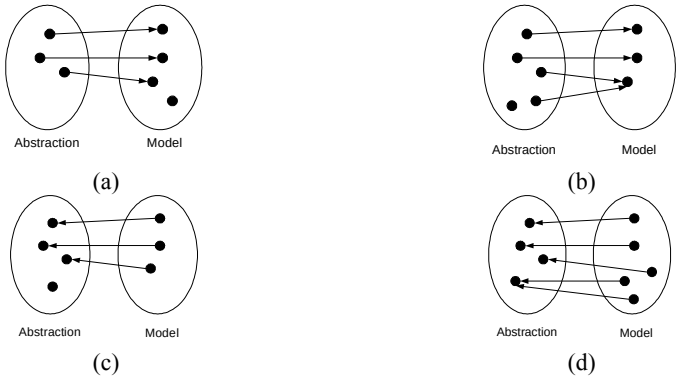

Figure 2. Examples of *Lucid* (a) and *Sound* (b) *representational* mappings from *Abstraction* to *Model*; Examples of *Laconic* (c) and *Complete* (d) *interpretation* mappings from *Model* to *Abstraction.*

2.1 Lucidity and Construct Overload

A model $\mathcal{M}$ is called *lucid* w.r.t. a domain abstraction $\mathcal{A}$ if a (representation) mapping from $\mathcal{A}$ to $\mathcal{M}$ is *injective*, i.e., iff every construct in the model $\mathcal{M}$ represents at most one (although perhaps none) concept of the domain 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 [3]. The ontological clarity of a modeling grammar is undermined by what is termed in [3] a *construct overload*. A construct overload occurs when a single language construct is used to represent two or more domain concepts. These notions albeit related are not identical. A construct can be overloaded at the language level, i.e., it can be used to model different concepts, but every manifestation of this construct in individual models is used to represent only one of the possible concepts. Figure 3 exemplifies a non-lucid representation. In this case, the construct X is used to represent two entities of the abstraction, namely the numbers 2 and 3. In this case, although the representation system does not have a case of construct overload (since labeled boxes only represent numbers and arcs only represent the less-than relation between numbers) the resulting model is non-lucid. In summary, the absence of construct overload in a language does not directly prevent

the construction of non-lucid representations in this language. Additionally, construct overload does not entail non-lucidity. Nevertheless, non-lucidity can also be manifested at a language level. We say that a language is non-lucid according to a conceptualization if there is a construct of the language that when used in a model of an abstraction (instantiation of this conceptualization) stands for more than one entity of the represented abstraction. Non-lucidity at the language level can be considered as a special case of construct overload that does entail non-lucidity at the model level.

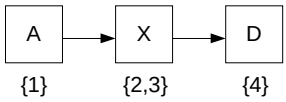

Figure 3. Example of a Non-Lucid Diagram**.**

Construct overload is an undesirable property of a modeling language since it causes ambiguity and, hence, undermines clarity. When a construct overload exists, users have to bring additional knowledge not contained in the model to understand the phenomena which is being represented. Additionally, a non-lucid representation language entails non-lucid representations which clearly violate the Gricean conversational maxim that requires contributions to be neither ambiguous nor obscure. 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. a domain abstraction $\mathcal{A}$ if a (representation) mapping from $\mathcal{A}$ to $\mathcal{M}$ is *surjective*, i.e., iff every construct in the model $\mathcal{M}$ represents at least one (although perhaps several) concept of the domain abstraction $\mathcal{A}$. An example of a surjective representation mapping is depicted in Figure 2(b).

An example of an unsound diagram is illustrated in Figure 4. The arc connecting the labeled boxes D and A does not correspond to any relation in the represented world. Unsoundness at the model level is strongly related to unsoundness at language level, a property that is termed *construct excess* in [3]. Construct excess occurs when a language construct does not represent any domain concept. Although construct excess results in the creation of unsound models, soundness at the language level does not prohibit the creation of unsound models. For example, there is no construct excess in the language used to produce the model of Figure 4.

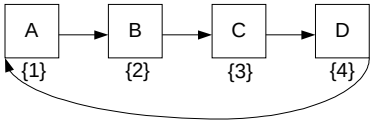

Figure 4. Example of an Unsound Diagram.

An unsound diagram violates the Gricean cooperative principle because any represented construct will be assumed to be meaningful by users of the language. Since no mapping is defined for the exceeding construct, its meaning becomes uncertain,

hence, undermining the clarity of the model. Users of 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 models they generate using the language [3]. 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. a domain abstraction $\mathcal{A}$ if an interpretation mapping between $\mathcal{M}$ and $\mathcal{A}$ is *injective*, i.e., iff every concept in the abstraction $\mathcal{A}$ is represented by at most one (although perhaps none) construct in the representation $\mathcal{M}$. An example of an injective interpretation mapping is depicted in Figure 2(c).

The notion of laconicity at the model level is related to the notion of *construct redundancy* at the language level in [3]. Construct redundancy occurs when more than one language construct can be used to represent the same domain concept. 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 models 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 models in that language. An example of a non-laconic diagram is illustrated in Figure 5. In this picture, the same domain entity (the number 3) is represented by two different constructs (C_1 and C_2) although the representation language used does not contain construct redundancy.

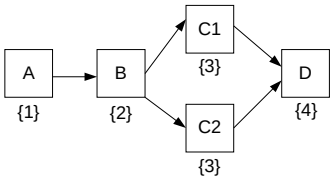

Figure 5. Example of a Non-laconic Diagram.

Non-laconicity can also be manifested at the language level. We say that a language is non-laconic if it has a construct that when used in a model of a domain abstraction, causes an entity of this abstraction to be modeled more than once in the resulting representation. For instance, take a version of the labeled boxes language used so far and let the less-than relation between numbers be represented both as the *transitive closure of the is-arrow-connected* and by the *is-smaller-than* relation between labeled boxes. All models using this representation (e.g., Figure 6) are deemed non-laconic. Non-laconicity at the language level can be considered as a special case of construct redundancy that does entail non-laconicity at the model level.

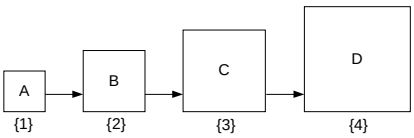

Figure 6. Example of a Non-laconic Diagram generated by a Non-Laconic Language**.**

In [3], the author claims that construct redundancy *"adds unnecessarily to the complexity of the modeling language"*, possibly confusing the users. Therefore, construct redundancy can also be considered to undermine representation clarity. Non-laconicity also violates the Gricean principle, since a redundant representation can be interpreted as standing for a different domain element. In sum, 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. a domain abstraction $\mathcal{A}$ if an interpretation mapping between $\mathcal{M}$ and $\mathcal{A}$ is *surjective*, i.e., if each concept in a domain abstraction (instance of the domain conceptualization) is represented by at least one (although perhaps many) construct in the representation $\mathcal{M}$. An example of a *surjective* interpretation mapping is depicted in Figure 2(d).

The notion of completeness at the model level 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 entails lack of expressivity, i.e., there can be phenomena in the considered domain that cannot be represented by the language. Alternatively, users of the language can choose to overload an existing construct in order to represent concepts that originally could not be represented, thus, undermining clarity. Thus, unless some existing construct is overloaded, an incomplete modeling language is bound to produce incomplete models. However, the converse is not true, i.e., a complete modeling language can still be used to produce incomplete models (see example in Figure 7). In Figure 7, a domain element (the $3 < 4$ relation) is omitted in the representation.

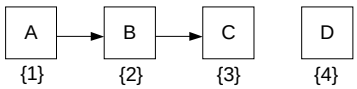

Figure 7. Example of an Incomplete Diagram**.**

In accordance with the detailed account of Grice's cooperative principle (specifically, that all necessary information is included), model and language designers should attempt to ensure completeness. In summary, a modeling language should be complete w.r.t. a domain conceptualization and every element in a domain abstraction (instance of this domain conceptualization) must be represented by an element of a model built using this language.

3 Conceptual Modeling, (Meta) Conceptualization and Ontology

According to Figure 1, a modeling language delimits all possible specifications[1] that can be constructed using this language, i.e., it determines all grammatically valid specifications of the language. Likewise, a conceptualization delimits all possible domain abstractions (representing state of affairs) that are admissible in that domain [5]. 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 is truthful to this domain if it 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, following [5], we present a formalization of this idea. This formalization compares conceptualizations as intentional structures and meta-models as represented by logical theories:

Let us first define a *conceptualization* C as an *intentional structure* $\langle W, D, \Re \rangle$ such that W is a (non-empty) set of possible worlds, D is the domain of individuals and $\Re$ is the set of n-ary relations (concepts) that are considered in C. The elements $\rho \in \Re$ are *intentional relations* with signatures such as $\rho^n: W \rightarrow \wp(D^n)$, so that each n-ary relation is a function from possible worlds to 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 apples in every possible world. For every world $w \in W$, according to C we have a *intended world structure* S_wC as a structure $\langle D, R_wC \rangle$ such that $R_wC = \{\rho(w) \mid \rho \in \Re\}$. More informally, we can say that every intended world structure S_wC 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 $S_c = \{SwC \mid w \in W\}$.

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 intentional interpretation by means of the structure $\langle C, \Im \rangle$, where $C = \langle W, D, \Re \rangle$ is a conceptualization and $\Im: V \rightarrow D \cup \Re$ is an intentional interpretation function which assigns elements of D to constant symbols in V, and elements of $\Re$ to predicate symbols of V. This intentional structure is named the *ontological commitment* of language $\mathcal{L}$ to a conceptualization C. A model $\langle S,I \rangle$ of $\mathcal{L}$ is said to be compatible with *ontological commitment* $K = \langle C, \Im \rangle$ if: (i) $S \in S_c$; (ii) for each constant c, $I(c) = \Im(c)$; (iii) there exists a world w such that for every predicate symbol p, I maps such a predicate to an admissible extension of $\Im(p)$, i.e. there is an intentional relation

[1] We have so far used the term *model* instead of specification since it is the most common term in conceptual modeling. In this section, 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.

ρ such that $\Im(p) = \rho$ 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 L according to K.

In order to exemplify these ideas let us take the example of a very simple conceptualization C such that W = {w,w'}, D = {a,b,c} and $\Re$ = {person, father}. Moreover, we have that person(w) = {a,b,c}, father(w) = {a}, person(w') = {a,b,c} and father(w') = {a,b}. This conceptualization accepts two possible state of affairs, which are represented by the world structures S_wC = {{a,b,c}, {{a,b,c},{a}} and $S_{w'}$C = {{a,b,c}, {{a,b,c},{a,b}}. Now, consider a language $\mathcal{L}$ whose vocabulary consists of the terms `Person` and `Father` with an underlying metamodel that poses no restrictions on the use of these primitives. In other words, the metamodel of $\mathcal{L}$ has the following logical rendering[2] (T_1): {∃x Person(x), ∃x Father(x)}. Clearly, we can 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'={a,b,c}, person = {a,b}, father = {c}, and I(`Person`) = person and I(`Father`) = father. This means that we can produce a specification using $\mathcal{L}$ which model is not an *intended model* according to C.

We now extend the metamodel of language $\mathcal{L}$ by adding one specific axiom and, hence, producing the metamodel (T_2): {∃x Person(x), ∃x Father(x), **∀x Father(x) → ∃x Person(x)**}. Contrary to $\mathcal{L}$, the resulting language $\mathcal{L}'$ with the amended metamodel 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 describes the set of all possible state of affairs that are considered admissible in the subject domain D. A representation O that has as *valid specifications* only those which represent *admissible state of affairs* according to conceptualization C is named an *Ontology* of domain D according to C. With an *explicit representation of a conceptualization* in terms of a *domain 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 valid models of the metamodel M of $\mathcal{L}$ and the set of valid models of the ontology O of D (see Figure 8). In the best case, these two specifications are *isomorphic* and, thus, they share the same set of logical models. Therefore, not only every entity in conceptualization C must have a representation in the metamodel M of language $\mathcal{L}$, but these representations must obey the same axiomatization.

In the example above, we address the domain of genealogical relations. This exemplifies what is named a *material domain* in the literature. Accordingly, a modeling language designed to represent phenomena in this domain is named a *Domain-Specific Modeling Language* [7]. However, we illustrate our approach here by considering a (domain-independent) *general conceptual modeling language* (e.g., EER, ORM, UML). What should be real-world conceptualization that this language should commit to? We argue that it should be a system of general categories and their ties, which can be used to articulate domain-specific common sense theories of reality. This meta-conceptualization should comprise a number of domain-independent theories (e.g., theory of parts and wholes, types and instantiation, identity, existential de-

[2] Given a model S in a modeling language $\mathcal{L}$, the logical rendering of S is defined as the logical theory T that is the first-order logic description of that specification [12].

pendence, etc.), which are able to characterize aspects of real-world entities irrespective of their particular nature. The development of such general theories of reality is the business of the philosophical discipline of *Formal Ontology* [8]. A concrete artifact representing the meta-conceptualizations is named a *Foundational Ontology* [9].

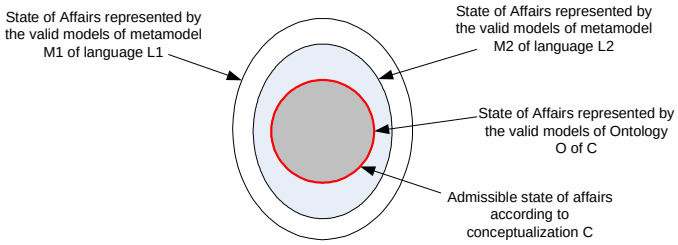

Figure 8. Measuring the degree of *domain appropriateness* of modeling languages via an ontology of a conceptualization of that domain.

4 The Unified Foundational Ontology (UFO-A)

In this section, we present a fragment of a philosophically and cognitively well-founded reference ontology (foundational ontology) that has been developed in [10, 12, 13, 14]. In particular, in [14], this ontology is named UFO (Unified Foundational Ontology) and is presented in three compliance sets. Here, we focus the first one (UFO-A), which is an *ontology of endurants*. In the sequel, we restrict ourselves to a fragment of UFO-A, depicted in Figure 9. Moreover, due to space limitations and the focus of the paper we briefly present the ontological categories comprising UFO-A (see aforementioned articles for details).

A fundamental distinction in this ontology is between the categories of ***Individual*** and ***Universal***. Individuals are entities that exist in reality possessing a unique identity. Universals, conversely, are space-time independent pattern of features, which can be realized in a number of different individuals. The core of this ontology exemplifies the so-called *Aristotelian ontological square* comprising the category pairs ***Substantial-Substantial Universal***, ***Moment-Moment Universal***. From a metaphysical point of view, this choice allows for the construction of a parsimonious ontology, based on the primitive and formally defined notion of *existential dependency*.

Definition 1 (existential dependence): We have that an individual *x* is *existentially dependent* of another individual *y* iff, as a matter of necessity, y must exist whenever x exists. ■

Existential dependence is a modally constant relation, i.e., if x is dependent of y, this relation holds between these two specific individuals in all possible worlds that x exists.

Substances are existentially independent individuals. Examples of Substances include ordinary mesoscopic objects such as an individual person, a dog, a house, a hammer, a car, Alan Turing and The Rolling Stones but also the so-called *Fiat Objects* such as the North-Sea and its proper-parts, postal districts and a non-smoking area of a restaurant.

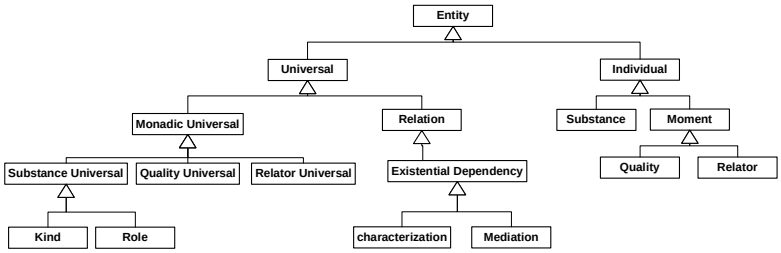

Figure 9. Excerpt of the Foundational ontology UFO-A.

The word ***Moment*** denotes, in general terms, what is sometimes named trope, abstract particular, individualized property or property in particular [9]. Therefore, in the scope of this work, the word bears no relation to the notion of time instant in colloquial language. A moment is an individual that can only exist in other individuals. Typical examples of moments are a color, a connection and a purchase order. Moments have in common that they are all dependent of other individuals (their bearers). Some moments are one-place ***Qualities*** (e.g., a color, a headache, a temperature); others are relational moments or ***Relators*** (e.g., a kiss, a handshake, a medical treatment, a purchase order), which depend on several substances.

A ***Substantial Universal*** is a universal whose instances are substances (e.g., the universal Person or the universal Apple). Within the category of substantial universals, we make a further distinction based on the formal notions of *rigidity* and *anti-rigidity*:

Definition 2 (Rigidity): A universal U is rigid if for every instance x of U, x is necessarily (in the modal sense) an instance of U. In other words, if x instantiates U in a given world w, then x must instantiate U in every possible world w'. ■

Definition 3 (Anti-rigidity): A universal U is anti-rigid if for every instance x of U, x is *possibly* (in the modal sense) not an instance of U. In other words, if x instantiates U in a given world w, then there must be a possible world w' in which x does not instantiate U. ■

A substantial universal which is rigid is named here a ***Kind***. In contrast, an anti-rigid substantial universal is termed a ***Role***. The prototypical example highlighting the modal distinction between these two categories is the difference between the universal (Kind) Person and the (Role) universal Student, both instantiated by the individual John in a given circumstance. Whilst John can cease to be a Student (and there were circumstances in which John was not one), he cannot cease to be a Person. In other words, in a conceptualization that models Person as a Kind and Student as a Role, while the instantiation of the role Student has no impact on the identity of an individual, if an individual ceases to instantiate the kind Person, then it ceases to exist as the same individual. Moreover, in [13], we have formally proved that a rigid universal cannot have as its superclass an anti-rigid one. Consequently, a Role cannot subsume a Kind in our theory.

A ***Quality Universal*** is a universal whose instances are individual qualities (e.g., the objectified color of this apple is an instance of the universal color), and a ***Relator Universal*** is one whose instances are individual relational moments (e.g., the particular enrollment connecting John and a certain University is an instance of the universal

Enrollment). Both quality and relator universals are moment universals. The relation between a substantial universal and quality universal is one of ***Characterization***. If a quality universal Q characterizes a substantial universal S, then every instance of Q is existentially dependent of an instance of S. Likewise, a relation between a set of substantial universals and a relator universal is one of ***Mediation***. If a relator universal R mediates the substantial universals $S_1 \ldots S_n$, then every instance of R is existentially dependent of a plurality of entities, namely, particular instances of $S_1 \ldots S_n$.

Relations are entities that glue together other entities. In the philosophical literature, two broad categories of relations are typically considered, namely, *material* and *formal* relations [15,16]. Formal relations hold between two or more entities directly, without any further intervening individual. The only formal relations considered in this article are the existential dependence relations aforementioned. Other examples include relations such as *part-of*, *subset-of*, *instantiation*, among others not discussed here [10]. Material relations, conversely, have material structure on their own and include examples such as kisses, conversations, fights and commitments. The relata of a material relation are mediated by relators. For example, an individual *purchase* is a relator that connects a customer and a supplier, and a *treatment* is a relator which connects a patient with a medical unit. The notion of relational moments is supported in several works in the philosophical literature (e.g., [15,16,17]) and, the position advocated here is that, relators play an important role in answering questions of the sort: what does it mean to say that John is married to Mary? Why is it true to say that Bill works for Company X but not for Company Y?

In this paper, we only countenance as relations those of existential dependency discussed above, i.e., characterization and mediation. Thus, by a relation here we mean a formal relation of existential dependency. Material relations are represented by explicitly representing their founding relators. Therefore, according to this theory, formal and material relations are entities of different ontological nature. Whilst a formal relation such as the one between John and his knowledge *x* of Greek holds directly and as soon as John and *x* exist, the relation of John being treated in a particular Medical Unit MU_1 is a contingent one, and must rely on the existence of a founding entity, such as, for instance, a treatment t in which both John and MU_1 participate.

5 Evaluating and extending UML for Conceptual Modeling

In this section we start by constructing representation and interpretation mappings between the concrete metaclasses of the UML metamodel presented in [18] and the ontological categories comprising the foundational ontology employed here.

We start our discussion by focusing on the meta-construct Class. We assume for now a specific notion of class, namely one whose instances are single objects (as opposed to tuples of objects). In this sense, the ontological interpretation of a UML Class is that of a *monadic universal*. However, by carrying on this process, we realize that in UML there are no modeling constructs that represent the leaf ontological categories specializing *monadic universal*, namely, *kind*, *role*, *quality* and *relator*. In other words, there are ontological concepts prescribed by our reference ontology that are not represented by any modeling construct in the language. This is a case of *construct incompleteness* at the modeling language level.

In UML, the association meta-construct is used to represent both formal and material relations. As discussed in Section 4, formal and material relations are considered here as entities belonging to disjoint ontological categories. Therefore, the representation mapping from both formal and material relations to associations in UML can be considered a case of *construct overload*. However, in a different perspective, there are refinements on the category of relations in UFO-A that have no representation in the UML metamodel (characterization and mediation). Here, we have another case of *construct incompleteness* at the modeling language level.

According to the UML specification, an interface is a declaration of a coherent set of features and obligations. It can be seen as a kind of contract that partitions and characterizes groups of properties that must be fulfilled by any instance of a classifier that implements that interface. In an interpretation mapping from the UML metamodel to the ontology of Figure 9, an interface qualifies as a case of *construct excess*. This means that since the UML interface is merely a design and implementation construct, there is no category in the conceptual modeling ontology proposed here that serve as the ontological interpretation for this construct.

In order to solve the cases of construct incompleteness in reference to the category of monadic universals, we propose a (lightweight) extension to the UML class meta-construct by introducing the stereotypes « kind », « role », « quality » and « relator », representing the respective ontological finer-grained distinctions. The profile formed by these newly introduced stereotypes must also contain a number of constraints that restrict the way the modeling constructs can be related. The goal is to have a metamodel such that all syntactically correct specifications using the profile have logical models that are *intended world structures* of the conceptualizations they are supposed to represent. Thus, for instance, in a conceptual model using this profile, a class stereotyped as « role » must include in its superclass collection one class stereotyped as « kind », since it is a postulate of our theory that anti-rigid universals cannot subsume rigid ones.

In general, qualities can be atomic or complex. Atomic quality universals are typically not represented in a conceptual model explicitly but via attribute functions that map each of their instances to points in a given *quality dimension*. For example, suppose we have the universal Apple (a substantial universal), characterized by the universal Weight. Thus, for an arbitrary instance *x* of Apple there is a quality *w* (instance of the quality universal Weight) that is existentially dependent of *x*. Associated with the universal Weight, and in the context of a given measurement system (e.g., the human perceptual system), there is a quality dimension *weightValue*, which is a set isomorphic to the half line of positive integers, obeying the same ordering structure. In this case, we can define an *attribute function* weight(Kg) which maps for every instance of apple (and in particular *x*) to a point in a quality dimension, i.e., its quality value. Due to space limitations we do not discuss here the case of atomic qualities and related notions[3]. A formal treatment of this subject can be found in [12].

An example of a complex quality universal is the universal *Symptom*, characterizing the role *Patient*: every individual Symptom is existentially dependent of an individual patient. Thus, even if the patients John and Paul experience headaches which

[3] We emphasize, nonetheless, that the same ontological concept of *attribute functions* is represented in the UML grammar both by the constructs of *attributes* and *navigable end names*, thus, amounting to a case of *construct redundancy* in the language.

are qualitatively indistinguishable, the headache of John is an individual which is only dependent of John. A complex quality universal is the ontological counterpart of the concept of *Weak entity types* in EER diagrams. We propose that they should be explicitly represented in class diagrams (via a class stereotyped as « quality »), or, to use an object-orientation term, *objectified*.

We advocate that associations in UML for the purpose of conceptual modeling should only represent formal relations. Consistently, we extend this construct in the UML metamodel by proposing the stereotypes « characterization » and « mediation » representing the two types of existential dependency considered here. Associations stereotyped as « characterization » must have in one of its association ends a class stereotyped as « quality » representing the characterizing quality universal.

In contrast, we propose to express relational properties explicitly via classes stereotyped as « relator », representing the ontological category of relator universals. The formal relation of mediation that takes place between the relator universal and the universals it mediates is explicitly represented by an association stereotyped as « mediation ». In addition, associations stereotyped as « mediation » must have in one of its association ends a class stereotyped as « relator ».

By representing relational properties explicitly via their founding relator universals, we not only remove the case of construct overload related to associations, but we also produce a representation that is more expressive, conceptually clear and semantically unambiguous. Consider, for example, the models depicted in Figure 10. In the standard UML representation of associations, the cardinality multiplicity of one-to-many between *GraduateStudent* to *Supervisor* is ambiguous and can be interpreted in a multitude of incompatible ways. For example, when stating that *"a supervisor supervises one to many student"* what exactly is being stated? (i) that in a given assignment there is one supervisor advising many students?, or (ii) that only one supervisor and one student are involved, but a supervisor can supervise many assignments? An analogous situation takes place when trying to interpret this association in the converse direction. In particular, due to the lack of expressivity of the traditional UML association, the model of Figure 10(a) cannot differentiate the two different conceptualizations, which are explicitly modeled in Figures 10(b) and 10(c). Finally, as discussed in [12], the problem of ambiguity of multiplicity constraints exemplified in these models only takes place in the case of material relations, in which two different types of constraints are collapsed.

Both *characterization* and *mediation* are directed relations. In the case of the former, the source is a quality universal, and in the case of the latter, the source is a relator universal. In both cases, the target is a substantial universal. Moreover, these two relations are mapped at the instance level to an *existential dependency* relation between the corresponding source individuals and their bearer objects. This has the following consequences for the extended UML metamodel: (i) the association end connected to the target (substantial) universal must have the cardinality constraints of one and exactly one, since every moment is a dependent entity; (ii) the association end connected to the target (substantial) universal must have the meta-attribute *isreadOnly = true*, since existential dependency is modally constant; and (iii) existential dependency relations are always binary relations. Finally, since a relator is dependent (mediates) on at least two numerically distinct entities, we have the following additional constraint: (iv) let R be a relator universal, let $\{C_1 \ldots C_2\}$ be a set of substantial univer-

sals mediated by R (related to R via a « mediation » relation) and let $lower_{Ci}$ be the value of the minimum cardinality constraint of the association end connected to C_i in the « mediation » relation, then $(\sum_{i=1}^{n} lower_{Ci}) \geq 2$.

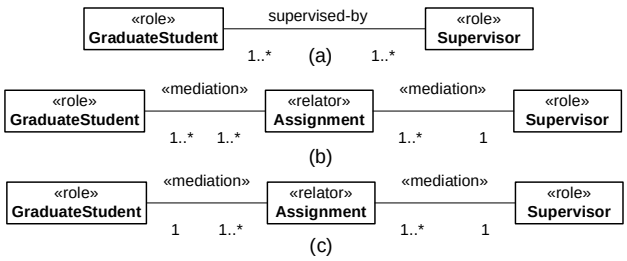

Figure 10.(a) Ambiguous representation of material relations using the standard UML notation **(b)(c)** Exemplification of how relators can disambiguate two conceptualizations that in the standard UML notation would have the same representation.

In order to solve the problem of construct excess in the case of UML interface metaclass, we propose to remove this construct from the extended UML metamodel, implying that the use of this construct would be prohibited in order to ensure that the resulting models are ontologically well-founded.

6 Final Considerations

In this paper, we present an ontology-based framework for evaluating the *domain* and *comprehensibility appropriateness* of modeling languages. The framework defines a systematic method for comparing the *metamodel* of a language with a concrete representation of a conceptualization of a given subject domain, termed a *reference ontology*. Moreover, the paper illustrates the application of the method by evaluating and extending a fragment of the *UML metamodel*. This has been achieved by comparing this metamodel with a *foundational ontology* that is considered as a suitable meta-conceptualization for domain independent *conceptual modeling*, and proposing stereotypes and usage constraints that make the metamodel isomorphic with the foundational ontology.

The framework presented here builds on existing work in the literature, extending them in important ways. For instance, the approaches of [2] and [3] 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, it does not consider the necessary mapping from these constraints to equivalent ones, to be established between the language constructs representing these ontological categories. Additionally, [3] addresses only the design of general conceptual modeling languages. In contrast, the framework and the principles proposed here can be applied to the design of conceptual modeling languages irrespective to each generalization level they belong, i.e., they can be applied both the level of material domains and corresponding domain-specific modeling languages, and the (meta) level of a domain-independent (meta) conceptualization that underpins a general conceptual modeling language. Finally, as discussed in [11], by explicitly

representing the subject domain of a language in terms of a well-founded ontology, we can account for important pragmatic aspects that should be preserved in the design of *concrete visual syntaxes*.

Acknowledgements. This work is part of the Freeband A-MUSE Project (contract BSIK 03025). We would like to thank Gerd Wagner, Nicola Guarino and Chris Vissers for fruitful discussions and for providing valuable input to the issues of this article.

References

1. Krogstie, J. (2001): *Using a Semiotic Framework to Evaluate UML for the Development of Models of High Quality*, Idea Publishing Group.
2. Gurr, C.A. (1999): *Effective Diagrammatic Communication: Syntatic, Semantic and Pragmatic Issues*, Journal of Visual Languages and Computing, 10, 317-342.
3. Weber, R. (1997). *Ontological Foundations of Information Systems*. Coopers & Lybrand, Melbourne.
4. Grice, H.P. (1975): *Logic and conversation*. In: Syntax and Semantics: Vol 3, Speech Acts (P. Cole & J. Morgan, eds). Academic Press, New York, pp. 43-58.
5. Guarino, N. (1998): *Formal Ontology in Information Systems*, Formal Ontology in Information Systems. Proceedings (FOIS), Italy.
6. Ciocoiu, M., Nau D. (2000): *Ontology-Based Semantics*. 7th International Conference on Principles of Knowledge Representation and Reasoning (KR'2000), USA.
7. Tolvanen, J.-P., Gray, J., Rossi, M., editors (2004): *Domain-Specific Modeling with Visual Languages*, Journal of Visual Languages and Computing, Elsevier Science.
8. Husserl, E. (1970): *Logical Investigations,* London: Routledge & Kegan Paul.
9. Masolo, C.; Borgo, S.; Gangemi, A.; Guarino, N.; Oltramari, A. (2003): *Ontology Library*, WonderWeb Deliverible D18.
10. Guizzardi, G., Herre, H., Wagner G. (2002): *On the General Ontological Foundations of Conceptual Modeling*, 21st Intl. Conf. on Conceptual Modeling (ER), Finland, LNCS 2503, Springer-Verlag.
11. Guizzardi, G.; Ferreira Pires, L.; van Sinderen, M. (2002): *On the role of Domain Ontologies in the Design of Domain-Specific Visual Languages*, 2nd Workshop on Domain-Specific Visual Languages, 17th OOPSLA, USA.
12. Guizzardi, G.; Wagner, G.; Herre, H. *On the Foundations of UML as an Ontology Representation Language*, 14th Intl Conference on Knowledge Engineering and Knowledge Management (EKAW), UK, LNCS 3257, Springer-Verlag.
13. Guizzardi, G.; Wagner, G.; Guarino, N.; van Sinderen, M. (2004): *An Ontologically Well-Founded Profile for UML Conceptual Models*, 16th Intl.Conference on Advances in Inf. Systems Eng. (CAiSE), Latvia, LNCS 3084, Springer-Verlag.
14. Guizzardi, G.; Wagner, G. (2005): *On a Unified Foundational Ontology and some Applications of it in Business Modeling*, Ontologies and Business Systems Analysis, Michael Rosemann and Peter Green (Eds.), IDEA Publisher.
15. Heller, B., Herre, H. *Ontological Categories in GOL*. Axiomathes 14: 71-90, Kluwer Academic Publishers, 2004.
16. Smith, B.; Mulligan, K (1986): *A Relational Theory of the Act*, Topoi (5/2), 115-30.
17. Schneider, L. (2003): *Designing Foundational Ontologies: The Object-Centered High-Level Reference Ontology OCHRE as a Case Study*, 22nd Intl. Conference on Conceptual Modeling (ER), LNCS 2813, Springer-Verlag.
18. Object Management Group (2003): *UML 2.0 Superstructure Specification*, Doc.# ptc/03-08-02, Aug. 2003.