Extending the Foundations of Ontology-based Conceptual Modeling with a Multi-Level Theory

Victorio A. Carvalho[1,2], João Paulo A. Almeida[1],
Claudenir M. Fonseca[1] and Giancarlo Guizzardi[1]

[1]Ontology & Conceptual Modeling Research Group (NEMO)
Federal University of Espírito Santo (UFES), Vitória, ES, Brazil
[2]Research Group in Applied Informatics, Informatics Department,
Federal Institute of Espírito Santo (IFES), Colatina, ES, Brazil
victorio@ifes.edu.br; jpalmeida@ieee.org; claudenirmf@gmail.com; gguizzardi@inf.ufes.br

Abstract. Since the late 1980s, there has been a growing interest in the use of foundational ontologies to provide a sound theoretical basis for the discipline of conceptual modeling. This has led to the development of ontology-based conceptual modeling techniques whose modeling primitives reflect the conceptual categories defined in a foundational ontology. The ontology-based conceptual modeling language OntoUML, for example, incorporates the distinctions underlying the taxonomy of types in the Unified Foundational Ontology (UFO) (e.g., kinds, phases, roles, mixins etc.). This approach has focused so far on the support to types whose *instances are individuals* in the subject domain, with no provision for *types of types* (or categories of categories). In this paper we address this limitation by extending the Unified Foundational Ontology with the MLT multi-level theory. The UFO-MLT combination serves as a foundation for conceptual models that can benefit from the ontological distinctions of UFO as well as MLT's basic concepts and patterns for multi-level modeling. We discuss the impact of the extended foundation to multi-level conceptual modeling.

Keywords: Ontology, Conceptual Modeling, Multi-level Modeling.

1 Introduction

Conceptual modeling is the activity of formally describing some aspects of the physical and social world around us for the purposes of understanding and communication [1]. It is generally considered a fundamental activity in information systems engineering [2], in which a given subject domain is described independently of specific implementation choices [3]. The main artefact of this activity is a conceptual model, i.e., a specification aiming at representing a conceptualization of the subject domain of interest.

Since the late 1980s, there has been a growing interest in the use of foundational ontologies to provide a sound theoretical basis for the discipline of conceptual modeling [4, 5, 6]. The initial hypothesis which was later confirmed by different empirical evidence can be explained by the following arguments: (i) conceptual models are

adfa, p. 1, 2011.

artifacts produced with the aim of representing a certain portion of reality according to a specific conceptualization; (ii) foundational ontologies describe the categories that are used for the development of these conceptualizations. Therefore, an appropriate conceptual modeling language should provide modeling primitives that reflect the conceptual categories defined in a foundational ontology. This observation has led to the development of approaches for conceptual modeling based on foundational ontologies. An example of such an approach is OntoUML, which is based on the Unified Foundational Ontology (UFO) [3]. In OntoUML, the taxonomy of types of the Unified Foundational Ontology (UFO) has been reflected in the language such that the distinctions of the foundational ontology can be used to provide useful constraints and modeling guidelines, ultimately leading to ontologically well-founded conceptual models. The resulting conceptual models consist of a collection of *types (classes) of individuals in the subject domain* (e.g., the "Person" kind, the "Child" phase, the "Student" role). Each of these domain types instantiate types in the foundational ontology (e.g., kind, subkind, role, phase, etc.).

The approach is so far unable to describe subject domains in which the categorization scheme itself is part of the subject matter. In these subject domains, experts make use of categories of categories in their accounts. For instance, considering the domain of human resource management, organizations are often staffed according to *employee types* (e.g. "Engineer", "Pilot", "Secretary"). They may need to classify those *employee types* giving rise to *types of employee types*. In this case, "Engineer" and "Pilot" could be considered as examples of "Technical Employee Type", as opposed to "Secretary" which is an example of "Administrative Employee Type". Finally, they need to track the allocation of personnel to specific departments (e.g. John is an engineer in the Maintenance Department). Thus, to describe the conceptualization underlying this domain, one needs to represent entities of different (but nonetheless related) classification levels, such as *individual persons* ("John"), *employee types* ("Engineer", "Pilot", "Secretary"), and *types of employee types* ("Technical Employee Type", "Administrative Employee Type").

The need to support the representation of subject domains that deal with multiple classification levels has given rise to what has been referred to as multi-level modeling [7, 8]. In order to address the challenge of multi-level modeling, we have proposed in [9] a theory called MLT. MLT formally characterizes the nature of classification levels, and precisely defines the relations that may occur between elements of different classification levels, encompassing different notions of power type [10, 11]. In this paper, we apply MLT to UFO, in order to extend its applicability to domains that require multiple levels of classification. Conceptual models built with the UFO-MLT combination benefit from the ontological distinctions of UFO as well as the basic concepts and patterns for multi-level modeling of MLT.

This paper is further structured as follows: section 2 presents a fragment of UFO and OntoUML; section 3 presents MLT; section 4 discusses the application of MLT to UFO identifying guidelines for multi-level modeling that arise from the foundational ontology; section 5 discusses the implications of the combined foundations to the practice of multi-level conceptual modeling and finally section 6 presents concluding remarks and topics for further investigation.

2 Ontological Foundations for Conceptual Models

The Unified Foundational Ontology (UFO) is a domain independent system of categories aggregating results from disciplines such as Analytical Philosophy, Cognitive Science, Philosophical Logics and Linguistics. Over the years, UFO has been successfully employed to analyze all the classical conceptual modeling constructs including Object Types and Taxonomic Structures, Part-Whole Relations, Intrinsic and Relational Properties, Weak Entities, Attributes and Datatypes, etc. [3]. Here we present a fragment of UFO that is relevant for this article. An in-depth discussion, formal characterization and discussion regarding empirical support for UFO's categories see [3].

UFO begins with a distinction between *universals* and *individuals*. Universals are patterns of features that can be realized in a number of individuals. For example, John and Mary are individuals that instantiate the universals "Man" and "Woman" respectively. UFO includes a taxonomy of individuals and a taxonomy of universals.

The topmost distinction in the taxonomy of individuals is that between endurants and events. Endurants (as opposed to events) are the individuals said to be wholly present whenever they are present, i.e., they can endure in time, suffering a number of qualitatively changes while maintaining their identity (e.g., a house, a person). Since in this paper we are especially interested in a portion of UFO that accounts for structural (as opposed to dynamic) aspects of conceptual modeling, we focus solely on endurants. Endurants are further classified into *Substantials* and *Moments*. Substantials are existentially-independent endurants (e.g. a person, a forest). A moment, in contrast, is an endurant that *inheres in*, and, therefore, is existentially dependent of, another endurant(s). Moments that are dependent of one single individual are *Intrinsic Moments* (e.g. a person's age) whereas moments that depend on a plurality of individuals are instances of *Relator* (e.g. a marriage, an employment, an enrollment).

These distinctions among individuals are reflected in the taxonomy of universals. Instances of *Intrinsic Moment Universal* apply to intrinsic moments (e.g. "Age"), instances of *Relator Universal* have relators as instances (e.g. "Marriage") and instances of "Substantial Universal" have substantials as instances (e.g. "Person"). The ontological category of Substantial Universal is further specialized according to the ontological notions of identity and rigidity. Substantial universals that carry a uniform principle of identity for their individuals are instances of *Sortal Universal* (e.g., "Person", "Car", "Organization"). In contrast, instances of *Non Sortal Universal* represent an abstraction of properties that are common to instances of various sortals (e.g., the non-sortal "Insurable Item" describes properties that are common to entities of different sortals such as "House", "Car", "Work of Art"). Moreover, a universal is said to be *rigid* if it classifies its instances necessarily (in the modal sense). In other words, if a universal T is rigid, then an instance x of T cannot cease to be an instance of T without ceasing to exist (e.g., "Person", "Organization"). In contrast, a universal is anti-rigid if its instances can move in and out of the extension of that universal without ceasing to exist (e.g., "Student", "Insured Organization"). Rigid sortals that provide a uniform principle of identity to their instances are termed a *Kind* (e.g "Person"). Instances of Kind may be specialized in other rigid sortals, which are instances of a *Subkind* (e.g. "Man"). Anti-rigid sortals are further classified into the categories *Role* or *Phase*. Instances of Role classify substantials through the relational properties they bear in the scope of a relational context (e.g. “Employee”, “Husband”, “Student”) whereas instances of Phase define partitions of a Kind depending on one or more of its intrinsic properties (e.g “Child”, “Living Person”). Rigid non-sortals that represent abstractions of properties that apply to instances of different kinds are called *Category* universals (e.g., “Legal Entity” abstracting properties of persons and organizations).

Fig. 1 summarizes the discussion so far by depicting a fragment of UFO’s taxonomy of universals in the left-hand side (“Endurant Universal” and its specializations) and the taxonomy of individuals in the right-hand side (“Endurant” and its specializations). This fragment of the UFO ontology is presented here as a UML class diagram for presentation-purposes only. The actual representation of the ontology is captured in [3] in a particular type of Intensional Modal Logics with Sortal Quantification.

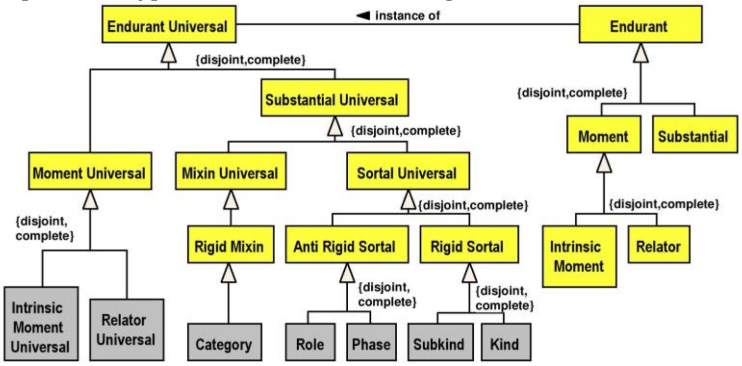

Fig. 1. UFO *endurant individuals* and *universals* taxonomies

In order to support the construction of ontology-driven conceptual models, a UML profile (dubbed OntoUML) was proposed in [3]. OntoUML includes: (i) modeling primitives that reflect ontological distinctions put forth by this ontology (these are represented as stereotypes for each of the leaf ontological categories of the UFO taxonomy of universals, see Fig. 2 for an example); (ii) formal constraints that govern how these constructs can be combined, which are derived from the axiomatization of the ontology. An example of constraint is that since instances of “Subkind”, “Role” and “Phase” carry the identity criteria provided by instances of “Kind”, OntoUML classes with a <<subkind>>, <<role>> or <<phase>> stereotype must specialize (directly or indirectly) exactly one class stereotyped as <<kind>>.

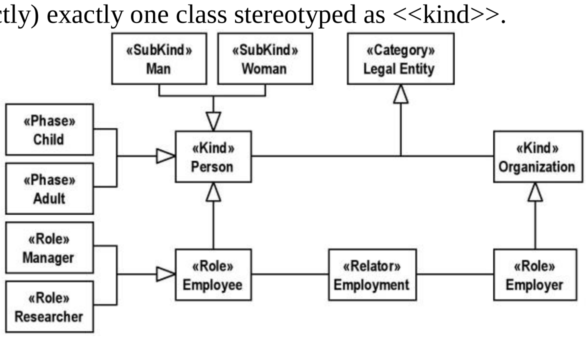

Fig. 2. An OntoUML diagram

3 MLT: A Theory for Multi-Level Modeling

Conceptual domain models constructed in OntoUML are able to express ontological properties of the types that apply to individuals in the subject domain. However, currently, no support is provided to represent domain-specific types of types, since the second-order types of OntoUML are predefined in the profile (as stereotypes). This motivates our investigation into the combination of UFO with a multi-level theory.

We employ a theory for conceptual multi-level called MLT which we introduced in [9]. Similar to UFO, MLT distinguishes between types (universals) and individuals. However, differently from UFO, MLT also considers types that have other types as instances. In order to accommodate these varieties of types, the notion of *type order* is used in MLT. Types having individuals as instances are called *first-order types,* types whose instances are first-order types are called *second-order types* and so on.

In order to link types to the entities that fall under such types, MLT defines a primitive *instance of* relation. This relation is represented by a ternary predicate *iof(e,t,w)* that holds if an entity *e* is instance of an entity *t* (denoting a type) in a world *w*. Indexing the instantiation relation to possible worlds allows MLT to support *dynamic classification*, admitting thus types that apply contingently to their instances (e.g., *John* is an instance of *student* in *w* but not in *w'*, when he has graduated.)

We build up the axiomatic theory defining the conditions for entities to be considered individuals, using the logic constant "Individual". Thus, an entity is an instance of "Individual" iff it does not have any possible instance. The constant "First-Order Type" (or shortly "1stOT") characterizes the type that applies to all entities whose instances are instances of "Individual". Analogously, each entity whose possible extension contains exclusively instances of "1stOT" is an instance of "Second-Order Type" (or shortly "2ndOT"). It follows from this definition that "Individual" is instance of "1stOT" which, in turn, is instance of "2ndOT". We call "Individual", "1stOT" and "2ndOT" the basic types of MLT. According to MLT, every possible entity must be instance of exactly one of its *basic types* (except the topmost type). For our purposes in this paper *first*- and *second-order types* are enough. However, this scheme can be extended to consider as many orders as necessary.

Fig. 3 illustrates the elements that form the basis for our multi-level modeling theory, using a notation that is largely inspired in UML. We use the UML class notation to represent the *basic types of* MLT. We use associations as usual to represent relations between instances of the related types (predicates that may be applied to instances of the related types). Since UML does not allow for the representation of links between classes, we use dashed arrows to represent relations that hold between the types, with labels to denote the names of the predicates that apply. This notation is used in all further diagrams in this paper. It is important to highlight here that our focus is not on the syntax of a multi-level modeling language and we use these diagrams to illustrate the concepts intuitively. A complete formalization of MLT can be found in [9].

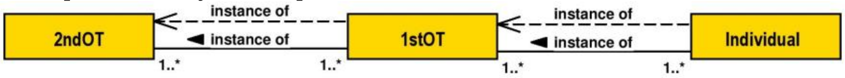

Fig. 3. Basic foundations of MLT: *basic types* and *instance of* relation.

Some structural relations to support conceptual modeling are defined in MLT, starting with the ordinary specialization between types. A type *t specializes* another type *t'* iff in all possible worlds all instances of *t* are also instances of *t'*. According to this definition every type *specializes* itself. Since this may be undesired in some contexts, we define the *proper specialization* relation as follows: *t proper specializes t'* iff *t specializes t'* and *t* is different from *t'*. Note that the definitions presented thus far guarantee that both *specializations* and *proper specializations* may only hold between types of the same order (these relations are depicted in the upper part of Fig. 4).

Every type that is not a basic type (e.g., a domain type) is an instance of one of the basic higher-order types (e.g., "1stOT", "2ndOT"), and, at the same time proper specializes the basic type at the immediately lower level (respectively, "Individual" and "1stOT"). Fig. 4 illustrates this pattern. Since "Person" applies to individuals, it is instance of "1stOT" and proper specializes "Individual". The instances of "Person Age Phase" are specializations of "Person" (e.g. "Child" and "Adult"). Thus, "Person Age Phase" is instance of "2ndOT" and proper specializes "1stOT". In section 4, this pattern will be applied to UFO concepts and will drive patterns for domain models.

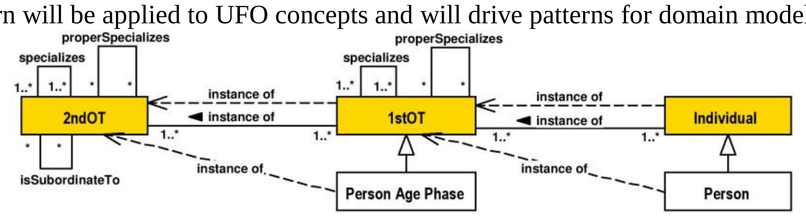

Fig. 4. Illustrating an important basic pattern of MLT and its intra-level structural relations.

In addition to the instantiation and specialization relations, MLT also defines a *subordination* relation. *Subordination* between two higher-order types implies *specializations* between their instances i.e., *t is subordinate to t'* iff every *instance of t proper specializes* an *instance of t'*. Since *subordination* implies *proper specializations* between the instances of the involved types at one order lower, subordination can only hold between higher-order types of equal order. We will use subordination to explain the relation between universals in UFO's taxonomy of universals (e.g., since every "Subkind" must specialize a "Kind", "Subkind" *is subordinate to* "Kind".)

So far, we have only considered intra-level relations, i.e., those that occur between entities of the same order. In addition to that, MLT defines *cross-level structural relations* between types of adjacent orders. These relations support an analysis of the notions of power type in the literature, leading to their incorporation in the theory.

Based on the notion of *power type* proposed by Cardelli [10] (which is founded on the notion of powerset), MLT defines a *power type* relation between a higher-order type and a base type at an order lower: a type *t is power type of* a base type *t'* iff all instances of *t specialize t'* and all possible *specializations of t' are instances of t*. For example, consider a type called "Person Powertype" such that all possible specializations of "Person" are instances of it and, conversely, all its instances specialize "Person". In this case, "Person Powertype" is the *power type* of "Person". Since "Person" is instance of "1stOT", "Person Powertype" is instance of "2ndOT" and specializes "1stOT" (see Fig. 5). (It follows from the definition of *power type* that "1stOT" is *power type of* "Individual". Analogously, "2ndOT" is *power type of* "1stOT".).

Another definition of *power type* that has had great influence in software engineering was proposed by Odell [11]. Inspired on Odell's definition [11], MLT defines the *characterization* relation between types of adjacent levels: a type *t characterizes* a type *t'* iff all *instances of t* are *proper specializations* of *t'*. Note that there may be specializations of the base type *t'* that are not *instances of t.* For instance in Fig. 5, "Person Role" (with instances "Manager" and "Researcher") *characterizes* "Person", but is <u>not</u> a *power type of* "Person", since there are specializations of "Person" that are not instances of "Person Role" ("Child" and "Adult" in the example).

We also define some variations of *characterization*, which are useful to capture further constraints in a multi-level model. We consider that a type *t completelyCharacterizes t'* iff *t characterizes t'* and every instance of *t'* is *instance of*, <u>at least</u>, an *instance of t.* Moreover, iff *t characterizes t'* and every instance of *t'* is *instance of*, <u>at most</u>, one *instance of t* it is said that *t disjointlyCharacterizes t'*. Finally, a common use for the notion of *power type* in literature considers a *second-order type* that, simultaneously, *completely* and *disjointly characterizes* a *first-order type*. To capture this notion we defined the *partitions* relation. Thus, *t partitions t'* iff each *instance of t* is *instance of* <u>exactly one</u> *instance of* the base type *t'.* For example of the partitioning relation, consider the second-order type called "Person Age Phase" with instances "Child" and "Adult" (Fig. 5). (This kind of constraint is usually represented in UML through a generalization set, see [9] for a detailed comparison).

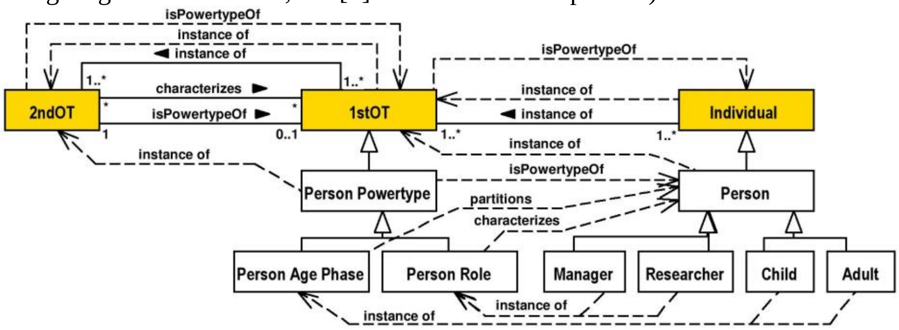

Fig. 5**.** Illustrating the use of MLT for multi-level conceptual modeling.

4 Combining MLT and UFO

We represent the MLT concepts in the topmost layer of a hierarchy of conceptual models. The basic pattern of MLT is applied in this hierarchy to establish the relation between MLT and UFO, and later to establish the relation between a conceptual domain model and UFO-MLT. More specifically, the concepts of UFO *instantiate* and *specialize* elements of MLT, thereby respecting MLT's axioms and leveraging the use of structural relations and patterns of MLT in UFO. In turn, the concepts of the conceptual domain models *instantiate* and *specialize* concepts of MLT and UFO, respecting all rules and patterns of both MLT and UFO.

The concepts in UFO's taxonomy of individuals are instances of "1stOT" specializing "Individual". The concepts in the taxonomy of universals are instances of

"2ndOT" specializing "1stOT". For each entity in the taxonomy of individuals (e.g., "Endurant", "Substantial", "Moment"), there is a corresponding entity in the taxonomy of universals ("EndurantUniversal", "SubstantialUniversal", "MomentUniversal"). Instances of the entity in the taxonomy of universals specialize the corresponding entity in the taxonomy of individuals. Thus, "EndurantUniversal" *characterizes* "Endurant", "SubstantialUniversal" *characterizes* "Substantial", and so on.

In addition to these general characterization relations, we can also use more specific MLT relations to further explain how the two taxonomies relate according to ground notions in UFO (such as identity). For example, since each instance of "Substantial" is an instance of exactly one instance of "Kind" (the kind that supplies the principle of identity), following MLT, "Kind" *partitions* "Substantial". In addition, since they carry (but do not supply) a principle of identity, instances of "Subkind", "Phase" and "Role" must specialize an instance of "Kind" that supplies such principle. Thus, in MLT terms, "Subkind", "Phase" and "Role" are *subordinate to* "Kind". Fig. 6 illustrates the resulting two-layer hierarchy revealing these relations.

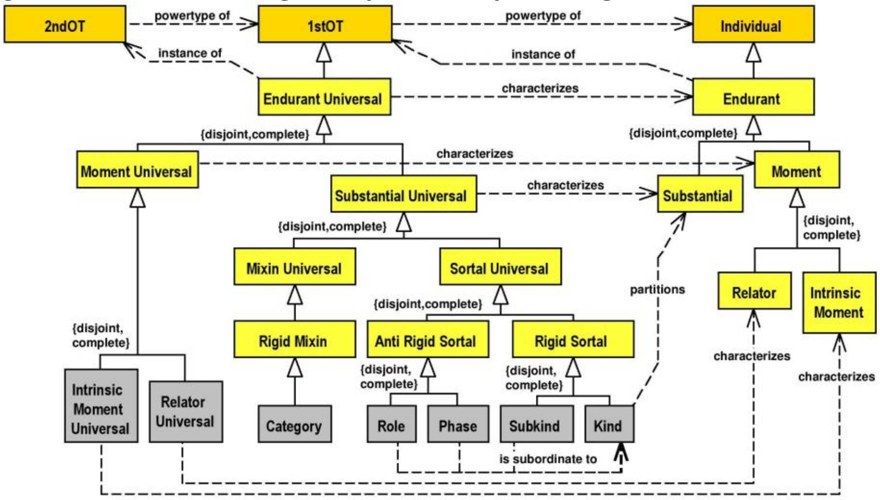

Fig. 6. Applying MLT to UFO taxonomies of endurants

In order to benefit from the ontological distinctions of UFO as well as the basic concepts and patterns for multi-level modeling of MLT, conceptual models built with the UFO-MLT combination must adhere to the rules of both theories. Thus, every domain *first-order* type must: (i) instantiate one of the leaf ontological categories of UFO's taxonomy of universals (and, consequently, instantiate MLT's "1stOT"); and (ii) simultaneously, specialize one of the leaf ontological categories of UFO's taxonomy of individuals (and thus, specialize "Individual"). For example, according to the conceptual domain model depicted in Fig. 2, "Legal Entity" is instance of "Category". Since "Category" specializes "1stOT" we conclude that "Legal Entity" is also a first-order type. Considering that "Category" *characterizes* "Substantial", it follows that "Legal Entity" specializes "Substantial" (and, indirectly, specializes "Individual"). Analogously, "Person" and "Organization" are instances of "Kind" and specialize "Legal Entity" (indirectly specializing "Substantial" and "Individual") (Fig. 7).

The UFO-MLT combination allows us to leverage the UFO taxonomy of universals to provide rules and patterns for *second-order types* in domain models. There are two basic rules for second-order types. First, since every domain first-order type admissible by UFO must be an instance of one of the leaf categories of the taxonomy of universals, *every domain second-order type must specialize one of these categories*. Second, to clarify which first-order type is ultimately instantiated by instances of a second-order type, *every domain second-order type must have an MLT cross-level relation with a first-order type (*i.e., *characterizes, disjointly characterizes, completely characterizes or partitions).* These rules are applied below for second-order domain types specializing different leaf categories of UFO's taxonomy of universals.

Specializations of Kind. Considering that instances of "Category" are generalizations of instances of "Kind", a second-order type that specialize "Kind" must *partition* an instance of "Category". For example, considering "Legal Entity" as a "Category" that generalizes properties of different kinds of legal entities, we may define a *second-order type* "Legal Entity Kind" that *specializes* "Kind" and *partitions* "Legal Entity" having as instances "Person" and "Organization". Fig. 7 illustrates this scenario.

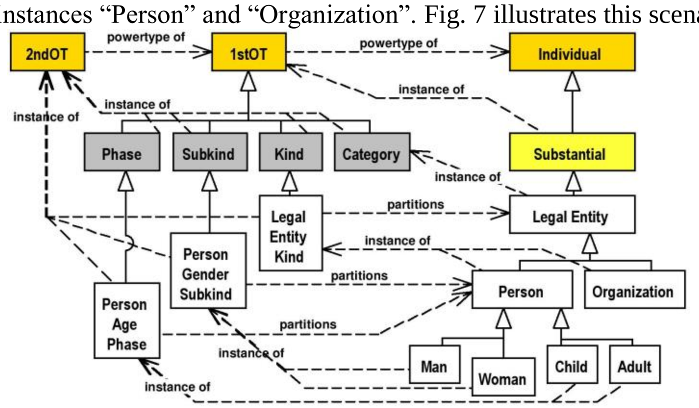

Fig. 7. Using the combination UFO-MLT to create a multi-level conceptual model

Specializations of Subkind. Every (instance of) "Subkind" is a rigid universal that specializes an instance of "Kind" according to some intrinsic properties exemplified by their instances. Thus, every *second-order type* that specializes "Subkind" must *characterize* an instance of "Kind". Subkinds are common in taxonomies in which the more specific types form a partition of a more general type distinguishing instances according to immutable intrinsic properties (e.g., "Person" specialized into "Man" and "Woman" according to gender). In this case, a *second-order type* that specializes "Subkind" *partitions* an instance of "Kind" according to immutable intrinsic properties exemplified by their instances (see "Person Gender Subkind" with instances "Man" and "Woman" in Fig. 7). Another example of *second-order type* that can be represented as a specialization of "Subkind" is "Dog Breed", *partitioning* "Dog". Note that MLT does not force the modeler to enumerate the instances of second-order types (such as "Dog Breed"), while still capturing the fact that its instances form a partition of the base type ("Dog").

Specializations of Phase. According to UFO, instances of "Phase" are anti-rigid types that specialize an instance of "Kind", "Subkind" or "Phase" according to some mutable intrinsic property. Thus, every *second-order type* that specializes “Phase” must *characterize* an instance of “Kind”, “Subkind” or “Phase”. As discussed in [3], the instances of “Phase” form partitions of a more general type. We can capture this notion with a *second-order type* that specializes “Phase” and *partitions* an instance of “Kind”, “Subkind” or “Phase” (e.g., in Fig. 7, “PersonAgePhase” *partitions* “Person” into “Child” and “Adult” according to age).

Specializations of Role. According to UFO, instances of “Role” are anti-rigid types that specialize an instance of a “Sortal Universal” (“Kind”, “Subkind”, “Phase” or another “Role”) classifying instances through the relational properties they bear in the scope of a relational context. Thus, every *second-order type* that specializes “Role” must *characterize* an instance of “Sortal Universal”. For example, we can define a *second-order type* “Person Role” that *characterizes* “Person” according to roles that persons may play during their lives. Types such as “Employee”, “Driver” and “Wife” would be examples of instances of “Person Role”. More specific specializations of “Person Role” include: (i) “Woman Role” whose instances specialize “Woman” (an instance of “Subkind”) and include those roles that are played exclusively by women, such as “Wife”; (ii) “Adult Role” whose instances specialize “Adult” (an instance of “Phase”) and include those roles that are played exclusively by adults, such as “Driver”; and, (iii) “Employee Role” whose instances specialize “Employee” (an instance of “Role”) and include those roles that are played exclusively by employees such as “Manager” and “Researcher”. These examples of second-order types specializing “Role” are illustrated in Fig. 8.

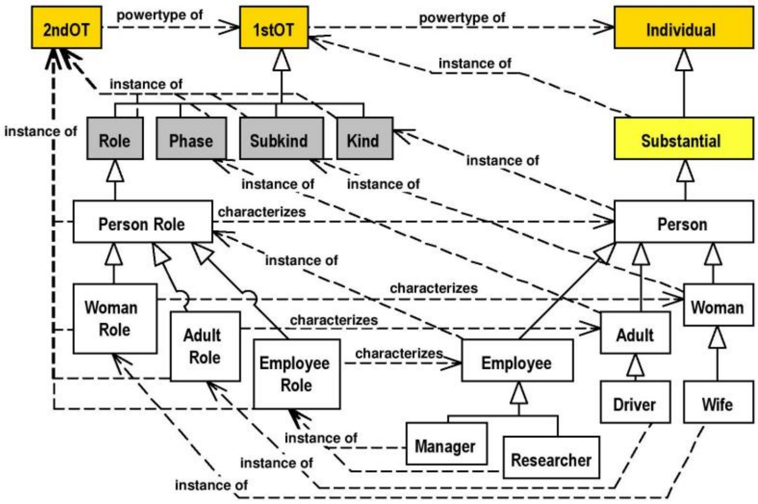

Fig. 8. Some patterns to create second-order types as specializations of “Role”

Note that the strategy that was used previously in OntoUML [3] was one in which the types represented in conceptual models could only instantiate the universals in UFO’s taxonomy of universals. These were represented by a fixed set of UML stereotypes, and thus a conceptual model could only have first-order types. In that approach, the axioms of the foundational ontology had to be incorporated into the syntax and semantics of the language profile (e.g., translated into corresponding syntactic rules as shown in [3], or incorporated in a transformation of OntoUML into a logical formalism). This additional step is not necessary here as the structural relations and axioms of UFO-MLT are directly incorporated in the domain ontology. For example, concerning the combinations of the specialization patterns for second-order types discussed in this section, it is inadmissible for a domain second-order type that specializes "Kind" and "Subkind" to be subordinate to a domain second-order type which specializes "Role" or "Phase". This is a consequence of the constraint in UFO that rules out the specialization of an anti-rigid universal by a rigid universal, together with the definition of subordination in MLT.

5 Related Work

Two early attempts to address multi-level modeling, namely *power types* [10, 11] and *materialization* [12], raised from the identification of patterns to represent the relationship between a class of categories and a class of more concrete entities. Despite their different origins, they are based on similar conceptualizations [13] addressing similar concerns. Both approaches establish a relationship between two types such that the instances of one are specializations of another. The power type approach was incorporated in the UML [14], and the language currently includes a *power type association* that relates a classifier (power type) to a generalization set composed by the generalizations that occur between the base classifier and the instances of the power type. Since OntoUML [3] does not add to UML any support for higher-order types, the only multi-level modeling support provided to OntoUML users is UML's support for *power types*. Because of its dependence on the generalization set construct, the UML power type pattern can only be applied when specializations of the base type are explicitly modeled (otherwise there would be no generalization set). We consider this undesirable, as it would rule out simple models that are possible in our approach, e.g., one defining "Dog Breed" as a power type of "Dog", without forcing the modeler to enumerate the instances of "Dog Breed". We capture the fact that the instances of "Dog Breed" form a partition of "Dog", regardless of their representation in a model. This is a more adequate choice considering our focus on representing a conceptualization as accurately as possible; this allows the representation of a conceptualization of dogs and dog breeds in general, without mention of specific dog breeds.

In [15], Erikson et al. propose a UFO-based approach that tries to avoid second-order types by employing a pattern based on the so-called *ontological square* comprising the categories of Substantial Univ./Substantial Individual and Moment Univ./Moment Individual, as well as their mutual relations. They provide an example in which "Horse" is considered a *substantial type*, a horse named "Prancer" is a *substantial object* (instance of "Horse"), "Breed" is a *moment type* and "Shetland Pony" is a *moment object* (instance of "Breed"). Since both *Prancer* and *Shetland Pony* are objects, there is no instance of relation between them. According to the authors, each instance of *Horse* is related to one instance of *Breed* and one instance of *Breed* is related to many instances of *Horse.* Their assumption that the same *moment object* can be related to various *substantial objects* is a misinterpretation of a basic rule of the foundational ontology. In UFO, the relation between *moment objects (individuals)* and *substantial objects (individuals)* is that of *inherence.* In the ontology literature, in general, and in UFO, in particular, it is not possible for an intrinsic moment to inhere in two different individuals. What the authors seem to intend to represent is actually the relation between a *property* (in the ontological sense) "Shetland Pony" and a number of individuals in which this property is *exemplified* (also in the ontological sense [3]). However, under this interpretation, "Shetland Pony" becomes a universal and "Breed" a second-order universal, defeating what they were trying to accomplish with their approach. Besides this ontological problem, the authors ignore the intuitive mechanisms of defining subtypes of a type according to properties of their instances and the benefits of such mechanisms. For example, using such approach, there is no support to represent properties that inheres only in instances of "Shetland Pony".

Atkinson and Kühne have proposed a deep instantiation based approach [8, 13] as a means to provide for multiple levels of classification whereby an element at some level can describe features of elements at each level beneath that level. The "potency" of an element defines how deep the instantiation chain produced by it may become. A "Mobile Phone Model" class is provided by the authors as an example of a class of potency of 2. The class is given an attribute "IMEI" also with potency of 2 meaning that instances of instances of "Mobile Phone Model" are assigned a value to the "IMEI" attribute. The authors consider that the main benefit of deep instantiation based approach is to support multi-level modeling without the need of introducing what they consider superfluous types (the required base type in the power type pattern) [13]. In the aforementioned example, the concept of "Mobile Phone" would be omitted from the domain model. While the deep instantiation approach can reduce the number of entities represented in a model, this strategy should be used with parsimony. This is because classes that instantiate higher-order classes "inherit" their properties with potency higher than one. In this case, the instantiation relation is overloaded with an implicit specialization relation, and semantic clarity is traded for reduction of model size. Further, by omitting a base type we become unable to express whether the instances of a higher-order type are disjoint types (i.e., we are unable to distinguish which form of *characterization* would apply). We are also prevented from determining metaproperties of the base type (such as e.g., rigidity). Note that the patterns that we have defined in section 4 address these issues specifically. For example, one could define "Mobile Phone Model" as a second-order type that specializes "Subkind" and partitions the first-order type "Mobile Phone" mean that mobile phone cannot instantiate two models and is an instance of the every same model throughout its existence.

The *multi-level objects* (or *m-objects)* [7] is another multi-level modeling approach that applies the notion of *deep instantiation*. This approach is based on the notion of objects that "encapsulate different levels of abstractions that relate to a single domain concept", the so-called *m-objects*. Considering the *mobile phone* example, the modeler could define an m-object named "Mobile Phone Model" with two levels of abstraction, namely *type* and *physical entity* levels. Properties that are characteristics of a mobile phone model, such as *screen size,* should be defined at the *type level* while the ones exemplified by mobile phones, such as *IMEI*, should be defined at the *physical entity level*. The approach defines a *concretize* relationship to associate different m-objects. For example, applying the *concretize relation* to “Mobile Phone Model” we could create an m-object named “IPhone6” attributing values to the properties defined at the type level of “Mobile Phone Model” and at the same time “inheriting” the properties defined at the physical entity level of “Mobile Phone Model” such as “IMEI”. Thus, the *concretize* relationship semantically overloads instantiation and specialization. Given that these are relations of different ontological nature, we believe this could affect the understandability and usability of the approach. Similarly to Atkinson and Kühne’s proposal [8, 13] the approach leads to a model with fewer elements, but prevents us from expressing important aspects of the first- and second-order types.

6 Conclusions and Future Work

In this paper we have extended the Unified Foundational Ontology with the MLT multi-level theory in order to provide foundations for ontology-based multi-level conceptual modeling. MLT is founded on the notion of (ontological) instantiation, which is applied regularly across levels (“orders”). An important basic pattern of the theory has influenced significantly our approach: types *instantiate* a type at an immediately-higher order and *specialize* the basic type of the order to which they belong.

We have shown how the elements of MLT can be used to serve as the topmost layer of a hierarchy of conceptual models, from a foundational ontology to conceptual domain models. The concepts of the foundational ontology *instantiate* and *specialize* elements of MLT, respecting its axioms and using structural relations and patterns of MLT. In turn, the concepts of the conceptual domain model *instantiate* and *specialize* UFO-MLT, respecting MLT and UFO axioms and patterns. The result is an approach to define conceptual domain models that can represent types as well as types of types while adhering to the rules of a foundational ontology.

UFO’s original taxonomy of (first-order) universals is leveraged in order to provide patterns for types of types in the domain model. These patterns guide the modeler in the definition of higher-order types and their relations allowing the modeler to express modal properties of instances of higher-order types. To the best of our knowledge, this is the first work to identify patterns and constraints for higher-order types based on a foundational ontology.

Another consequence of employing MLT concerns the engineering of UFO itself. UFO’s taxonomies can now be explained in terms of instantiation of higher-order types. Further, as shown in section 4, the relations of MLT (such as characterization) can be used to explain how elements in the taxonomy of universals relate to elements in the taxonomy of individuals.

While we have focused in the definition of domain models, the approach discussed here forms the basis for further extension of UFO itself, as well as to include core ontologies in the hierarchy of models between the foundational ontology and domain models. We will apply this approach to improve the formalization of UFO-based ontologies whose conceptualizations span multiple levels of classifications (e.g., the UFO-S core ontology for services [16] and the O3 organizational ontology [17]).

Finally, we should stress that it is not our intention in this paper to propose a multi-level language, and that our use of a notation inspired in UML has been solely illustrative. As discussed in [3], a reference ontology can be used to inform the revision and redesign of a modeling language, not only through the identification of semantic overload, construct deficit, construct excess and construct redundancy, but also through the definition of modeling patterns and semantically-motivated syntactic constraints. Thus, a natural application for UFO-MLT is to inform the design of a well-founded multi-level conceptual modeling language or to promote the redesign of a language such as UML into a multi-level modeling language.

Acknowledgements. This research is funded by the Brazilian Research Funding Agencies CNPq (grants number 311313/2014-0, 485368/2013-7 and 461777/2014-2) and CAPES/CNPq (402991/2012-5). Victorio A. Carvalho is funded by CAPES.

References

1. Mylopoulos, J.: Conceptual Modeling and Telos. In: Loucopoulos, P., Zicari, R. (Eds.), Conceptual modeling, databases and CASE, pp. 49-68, Wiley (1992)
2. Olivé, A.: Conceptual Modeling of Information Systems. Springer (2007)
3. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. University of Twente, Enschede, The Netherlands (2005)
4. Wand, Y., Weber, R.: An ontological evaluation of systems analysis and design methods. In: Falkenberg E., Lingreen, P. (Eds.), Information System Concepts: An In-Depth Analysis. Elsevier Science Publishers B.V., North-Holland (1989)
5. Wand, Y. and Weber, R.: On the ontological expressiveness of information systems analysis and design grammars. Journal of Information Systems, (3), 217-237 (1993)
6. Guarino, N.: Formal Ontology and Information Systems. In Guarino, N. (ed.), Formal Ontology in Information Systems, Proc. FOIS 1998. IOS Press, Amsterdam: pp. 3-15 (1998)
7. Neumayr, B., Grün, K., Schrefl, M.: Multi-level domain modeling with m-objects and m-relationships. In: Proc. 6th Asia-Pacific Conf. on Conceptual Modeling, New Zealand (2009)
8. Atkinson, C., Kühne, T.: The Essence of Multilevel Modeling. In: Proc. Of the 4th International Conference on the Unified Modeling Language. Toronto, Canada (2001)
9. Carvalho, V. A., Almeida, J. P. A.: Towards a Well-Founded Theory for Multi-Level Conceptual Modeling. Submitted (2015), available at http://nemo.inf.ufes.br/mlt
10. Cardelli, L.: Structural Subtyping and the Notion of Power Type. In Proc. Of the 15th ACM Symposium of Principles of Programming Languages, pp. 70-79 (1988)
11. Odell, J.: Power types. In: Journal of Object-Oriented Programing, 7(2), pp. 8-12.(1994)
12. Pirotte, A., Zimanyi, E., Massart, D., Yakusheva, T.: Materialization: a powerful and ubiquitous abstraction pattern. Proc. 20th Int. Conf. Very Large DataBases, pp. 630–641 (1994)
13. Atkinson, C., Kühne, T.: Reducing accidental complexity in domain models. Software & Systems. Modeling, 7(3), pp 345-359. Springer-Verlag (2008)
14. OMG : UML Superstructure Specification – Version 2.4.1 (2011)
15. Eriksson. O., Henderson-Sellers, B., Ågerfalk, P. J.: Ontological and linguistic metamodeling revisited: A language use approach. Information and Software Technology, 55(12), pp. 2099-2124. Elsevier (2013)
16. Nardi, J. C., Falbo, R., Almeida, J. P. A., Guizzardi, G., et al.: A commitment-based reference ontology for services. Information Systems, 51, p.1 (2015)
17. Pereira, D., Almeida, J. P. A.: Representing Organizational structures in an enterprise architecture language. In 6th Workshop on Formal Ontologies meet Industry (2014)