On the Foundations of UML as an Ontology Representation Language

Giancarlo Guizzardi[1], Gerd Wagner[2], and Heinrich Herre[3]

[1]Centre for Telematics and Information Technology,
University of Twente, Enschede, The Netherlands
guizzard@cs.utwente.nl
[2]Eindhoven Univ. of Technology, Faculty of Technology Management,
G.Wagner@tm.tue.nl
[3]Institut fur Informatik, University of Leipzig and
Ontologies in Medicine Research Group (OntoMed)
herre@informatik.uni-leipzig.de

Abstract. There is a growing interest in the use of UML class diagrams as a modeling language to represent domain ontologies. In a series of papers [1,2] we have been using the General Ontological Language (GOL) and its underlying foundational ontology, proposed in [3,4], to evaluate the ontological correctness of a conceptual UML class model and to develop guidelines for how the constructs of the UML should be used in conceptual modeling and ontology representation. This paper can be seen as a continuation of this work, in which we focus on analyzing the UML metaconcepts of classes, attributes, data types and associations from an ontological point of view.

1 Introduction

The Unified Modeling Language (UML) is a language initially proposed as a unification of several different visual notations and modeling techniques used for systems design [5]. UML is now a *de facto* standard for modeling computational systems, and has recently been proposed that the language should be also used as an Ontology Representation Language [6]. A more explicit statement of interest on applying UML for ontology representation is made by the OMG Ontology Metamodel Definition Request for Proposals [7].

While modeling languages such as UML are evaluated on the basis of their successful use in information systems development, ontology specification languages and their underlying upper level ontologies have to be rooted in principled philosophical theories about what kinds of things exist and what their basic relationships with each other are. We believe that defining UML constructs only in terms of its mathematical semantics, although essential, is not sufficient to make it a suitable ontology representation language. We claim that, in order to model reality, a modeling language should be founded on formal upper-level ontologies, i.e it should have both formal and ontological semantics.

In a series of papers (e.g., [1,2]), we have used philosophical and psychological well-founded theories to evaluate the ontological correctness of UML conceptual models, and to develop methodological tools (language extensions, guidelines, profiles and design patterns) that assign principled ontological semantics to UML model-

ing constructs. In [1], we focus on the analysis of the representation of *part-whole* (*mereological*) relations in UML and propose necessary extensions to the language to represent ontologically distinct sorts of parthood. While in [2], we present a theory of universals, which are then used to propose a *UML profile* representing different types of classifiers. This paper is a continuation of this work, focusing on the most basic ontology representation constructs, namely *classes*, *attributes*, *data types* and *associations*. Our main objective is to use a philosophically sound foundational ontology to provide: (i) an interpretation of UML constructs in terms of ontological categories and relations; (ii) principles for how these constructs should be used for ontology representation; (iii) guidelines to evaluate the ontological correctness of a conceptual UML class model representing a domain ontology.

The remaining of this article is structured as follows: section 2 presents a selection of concepts from a foundational ontology that form the theoretical basis of the analysis conducted in this paper. The theory is further used in section 3 to examine the UML metaconcepts of class, attributes, data type and associations from an ontological point of view. Section 4 provides some final considerations.

2   Background: Ontological Categories

Figure 1 depicts some of the basic elements of the foundational ontology, which will be referenced throughout this work. This ontology represents a version of the General Formal Ontology (GFO) underlying the language GOL as presented in [3,4]. The General Ontology Language (GOL) and GFO are under development as part of the OntoMed (Ontologies in Medicine) Research Group at the University of Leipzig. The GOL project was launched in 1999 as a collaborative effort between philosophers, linguists and other cognitive scientists and computer and information scientists with the aim to, on one hand, construct a formal framework for building models and representing complex structures of the world, and, on the other hand, at the development and implementation of domain-specific ontologies in several fields, especially medical sciences. GFO has been proven insightful in providing a principled foundation for analyzing and extending conceptual modeling and ontology representation languages and constructs [1,8].

Due to our objectives in the scope of this paper, we focus our discussion on a limited subset of the categories comprising GFO, which are briefly explained in the following subsections. For a complete and detailed presentation of GOL and GFO, one should refer to [3,4]. For a comparison between GFO and other upper-level ontologies, such as the IEEE Standard Upper Ontology, KIF, Sowa, Russel and Norvig and LADSEB (that can be considered a preliminary version of DOLCE [10]), one should refer to [3].

2.1   Sets and Urelements

A fundamental distinction in this ontology is between the categories of urelements and sets. We assume the existence of both urelements and sets in the world. Urelements are entities which are not sets. They form an ultimate layer of entities without any
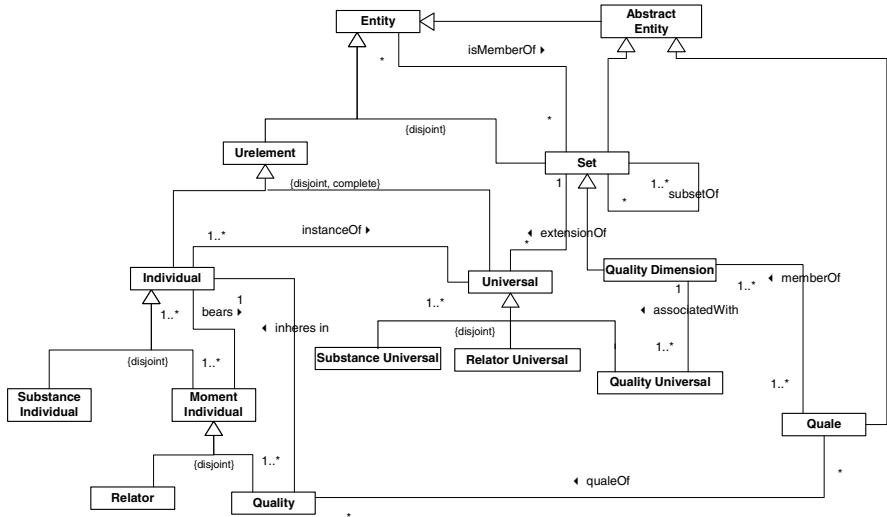
Fig. 1. Taxonomy of a fragment of GFO as a UML/MOF metamodel

set-theoretical structure in their build-up. Neither the membership relation nor the subset relation can unfold the internal structure of urelements. In GFO, urelements are classified into two disjoint categories of individuals and universals. Individuals are further classified in substance and moments.

2.2 Substance

Substances are individuals that can exist by themselves; this implies that a Substances is existentially independent from other individuals. Existential independence was introduced by E. Husserl: An individual A is existentially independent from an individual B if and only if it is logically possible for A to exist even if B does not exists. Examples of Substances include ordinary mesoscopic objects such as a dog, a house, a hammer, a car, Alan Turing and The Rolling Stones but also the so-called Fiat Objects [11] such as the North-Sea and its proper-parts, postal districts and a non-smoking area of a restaurant.

2.3 Moment

The word *Moment* is derived from the german *Momente* in the writings of Husserl and it denotes, in general terms, what is sometimes named *trope*, *abstract particular*, or *individualized property*. Therefore, in the scope of this work, the word bears no relation to the notion of time instant in ordinary parlance. *Moments* are individuals, which can only exist in other individuals (in the way in which, for example an electrical charge can exist only in some conductor). In other words, we can say that moments are *existentially dependent* on other individuals. The category of moments

includes: (i) *qualities*: moments that are dependent on one single individual (an individual color or weight, an electric charge); (ii) *relators* (or relational moments): existentially dependent on a plurality of individuals such as a kiss, a handshake, a covalent bond, but also social objects such as a flight connection, a purchase order and a commitment or claim. The inherence relation i – sometimes called ontic predication – glues moments to the substances which are their bearers. For example it glues your smile to your face, or the charge in a specific conductor to the conductor itself. In our framework we adopt the so-called adopt non-migration principle [3]: it is not possible that an intrinsic moment m inheres in two different substances a and b. As a consequence, if we have two particular substances a (a red apple) and b (a red car), and two moments m_1 (particular redness of a) and m_2 (particular redness of b), we consider m_1 and m_2 to be different individuals, although perhaps qualitatively indistinguishable. What does it mean then to say that a and b have the same color? In conformance with DOLCE [10], we distinguish between the color of a particular apple (its quality) and its 'value' (e.g., a particular shade of red). The latter is named *quale*, and describes the position of an individual quality within a certain *quality dimension*. The notions of *quality dimension* is discussed as follows.
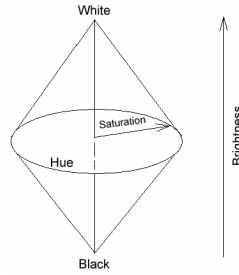
2.4   Quale, Quality Dimension and Quality Domain

An attempt to model the relation between qualities and their representation in human cognitive structures is presented in the theory of conceptual spaces developed by the Swedish philosopher and cognitive scientist Peter Gardenfors [9]. The theory is based on the notion of *quality dimension*. The idea is that for each perceivable or conceivable quality type there is an associated quality dimension in human cognition. For example, *height* and *mass* are associated with one-dimensional structures with a zero point isomorphic to the half-line of nonnegative numbers. Other qualities, such as color and taste, are represented by several dimensions. For instance, taste can be represented as tetrahedron space comprising the dimensions of saline, sweet, bitter and sour.

Gardenfors distinguishes between *integral* and *separable* quality dimensions: *"certain quality dimensions are integral in the sense that one cannot assign an object a value on one dimension without giving it a value on the other. For example, an object cannot be given a hue without giving it a brightness value…Dimensions that are not integral are said to be separable, as for example the size and hue dimensions."* [9, p. 24]. He then defines a *quality domain* as *"a set of integral dimensions that are separable from all other dimensions"* [9, p. 26]. Finally, he defends the idea that quality domains are endowed with certain structures (topological or ordering structures) that constrain the relations between its constituting dimensions. In his framework, the perception or conception of a quality individual can be represented as a point a in a quality domain. In accordance with DOLCE [10], this point is named here a *quale*.

An example of a quality domain is the set of integral dimensions related to color perception. A color quality c of an apple a takes its value (quale) in a three-dimensional color domain constituted of the dimensions hue, saturation and brightness. Figure 2 depicts the geometric space generated by the three quality dimensions that form this domain. One should notice that this structure constraints the relation

between some of these dimensions. In particular, saturation and brightness are not totally independent, since the possible variation of saturation decreases as brightness approaches the extreme points of black and white, i.e., for almost black or almost white, there can be very little variation in saturation.

The position defended here is that the notion of a quality domain (and the constraints relating different quality dimensions captured in its structure) can provide a sound basis for the domain ontology representations of the corresponding quality universal, constraining the possible values that its attributes can assume. This point is discussed and illustrated in section 3.2. We adopt here as a quality domain any collection of integral quality dimensions regardless if these dimensions are physically, theoretically or socially motivated.

Fig. 2. The quality dimensions of hue, saturation and brightness forming the color splinter [9]

Gardenfors also advocates that, from a metaphysical point of view, quality dimensions and the relations between them as well as quality domains are *"theoretical entities that can be used to explain and predict various empirical phenomena concerning concept formation"* [9, p. 31], i.e., abstract entities. For the purpose of this article we also take qualia to be abstract entities and represent quality dimensions as sets of *qualia* (see fig. 1). For instance, the *mass* dimension can be represented as a subset the set of Real numbers (respecting the same axiomatization) and the *hue* dimension can be represented as an enumeration of color qualia augmented with a set of formal relations between its member qualia (e.g. *complementaryOf* and *closeTo*). A quality domain is thus defined as a subset of the cross-product between its constituent integral quality dimensions (e.g. ColorDomain $\subset$ HueDimension $\times$ SaturationDimension $\times$ BrightnessDimension). The formation rule for the tuples that are members of a quality domain must obey the constraints that relate its quality dimensions.

We adopt here the formal relations *assoc(x,y)*, with the meaning "quality dimension x is associated with quality universal y, and, *ql(x,y)* that represents that relation between a quality individual *y* and its quale *x*. Among others, the following axiom is stipulated for these relations: if *x* is the quale of *y* then *x* must be a member of the quality dimension associated with the universal of which *y* is an instance or, formally, $\forall x,y \; ql(x,y) \rightarrow \exists z \; (y::z) \wedge \exists w \; assoc(w,z) \wedge (x \in w).$

In sum, if we have two particular substances *a* (a red apple) and *b* (a red car), and two moments m_1 (particular redness of *a*) and m_2 (particular redness of *b*). When saying that *a* and *b* have the same color, we mean that their individual color qualities

m_1 and m_2 are different, however, they can both be mapped to same point in the color quality domain, i.e., they have the same quale. The relation between a substance, one of its qualities and the associated quale is illustrated in figure 3.
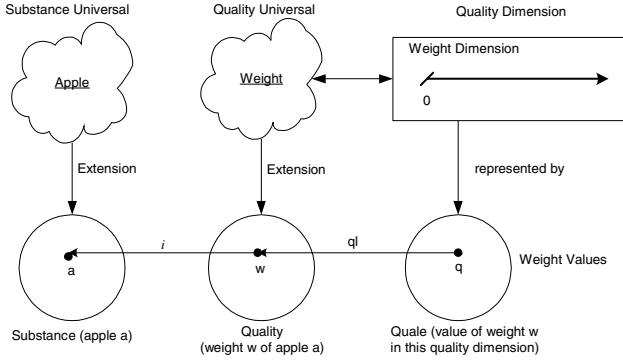

Fig. 3. Substances, qualities and qualia

2.5   Universals

A universal is a space-time independent pattern of features, which can be realized in a number of different individuals. Every individual instantiate at least one universal. Consequently, we account for the existence of Substantial Universals, Quality Universals, Relational Universals and so forth.

 We use the symbol :: to denote the instantiation relation, a basic formal relation defined to hold between individuals (first argument) and universals (second argument). Hence, when writing x::U we mean that x is an instance of U or that x has the property of being a U. For example, x can be a molecule of DNA and *u* a pattern of features shared by all exactly similar molecules, where the notion of exact similarity is determined by the granularity and point of view of genetic science.

 In this article, two universals which of particular interest are ***quality universals***, such as color and weight, and ***relational universals***, such as flight connection ('…is connected with…') or purchase ('…purchases…from…'). Every universal has an intension which, in our approach, is captured by means of an axiomatic specification, i.e. a set of axioms that may involve a number of other universals representing its essential properties. A particular form of such a specification of a universal U, called elementary specification, consists of a number of universals $U_1,...,U_n$ and corresponding functional relations $R_1, ...,R_n$ which attach instances from the U_i to instances of U, expressed by the following axiom:

$$\forall a\ (a::U \rightarrow \exists e_1...\exists e_n \bigwedge_{i \leq n} (e_i ::U_i \wedge R_i(a,e_i)))$$

 The universals $U_1,...,U_n$ used in an elementary specification are called features. A special case of an elementary specification is a quality specification where $U_1,...,U_n$

are quality universals, the instances of U are substances and R_i represents the formal inherence relation i.

2.6 Relations and Relators

Relations are entities which glue together other entities. We divide relations into two broad categories, called *material* and *formal*. Formal relations hold between two or more entities directly without any further intervening individual. Examples of formal relations are: 5 *is greater than* 3, this day is *part-of* this month, and N *is subset of* Q but also the relations of instantiation (::), inherence (*i*), quale of a quality (*ql*), *assoc*, *dependence*, among others. In principle, the category of formal relations includes those relations that form the mathematical superstructure of our framework [1,3]. However, we also classify as formal those domain relations that exhibit similar characteristics, i.e. those relations of comparison such as *is taller than*, *is older than*, *know more greek than*. As pointed out in [12], the entities which are immediate relata of such relations are not substances but moments. For instance, the relation *heavier-than* between two atoms is a formal relation which holds directly as soon as the relata (atoms) are given. The truth-value of a predicate representing this relation depends solely the atomic number (intrinsic moment) of each atom and the material content of *heavier-than* is as it were distributed between the two relata.

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 individuals which are called *relators*. Relators are individuals with the power of connecting entities; a flight connection, for example, is a relator that connects airports, an enrollment a relator that connects a student with an educational institution. For the purpose of this article we define a relator simply as an *individual* r which is one-sidedly existentially dependent on two or more individuals x and y, discrete from r and from each other; r is said to *relate* x to y [12]. The notion of relators is supported in several works in the philosophical literature [4,8,12] and, the position advocated here is that, relators play an important role in answering questions of the sort: what does it mean say that John is married to Mary? Why is it true to say that Bill works for Company X but not for Company Y?

A relator universal is a universal whose instances are relators. If r connects the entities $a_1,\ldots,a_n$, then this yields a new individual which is denoted by $\langle r: a_1,\ldots,a_n \rangle$. Individuals of this latter sort are called material facts. For every relator universals R there exists a set of facts, denoted by *facts*(R), which is defined by the instances of R and their corresponding arguments. We assume the axiom that for every relator universal R there is a factual universal F(R) whose extension equals the set *facts*(R). The factual universal F(R) is the basis for the material relation R(F) whose instances are n-tuples of entities.

In general, a relation universal R(F) can thus be defined as follows. Let $\phi(a_1,\ldots,a_n)$ denote a condition on the individuals $a_1,\ldots,a_n$

$$[a_1\ldots a_n]: R(F)\,(U_1\ldots U_n) \leftrightarrow \bigwedge_{i\,\leq\,n} a_j{::}U_i \wedge \phi\,(a_1\ldots a_n)$$

A relation is called *material* if there is a relator universal *R* such that the condition ϕ is obtained from R as follows: $\phi(a_1\ldots a_n) \leftrightarrow \exists k\,(k{::}R \wedge \langle k{:}a_1\ldots a_n\rangle{::}F(R)).$

Otherwise, R(F) is a formal relation whose instances are formal facts of the form $\langle R(F):a,b \rangle$.

Example: Let *Conn* be binary relator universal whose instances are individual flight connections. Then we may form a factual universal $F_{Conn} = F(Conn)$ having the meaning "An airport *X* is connected to a airport *Y*" whose instances are all facts of the form $\langle c: a,b \rangle$ where c is an individual flight connection and a,b are individual airports. A relation universal $R_{connected-to} = R(F_{Conn})$ is defined as such that its instances are pairs of the form [a,b].

3   Ontological Foundations for UML Class Diagrams

In the sequel, we refer to the *OMG UML Superstructure Specification 2.0* [5], when quoting text in italics. For simplicity, we write *UML-ontology* when we mean domain ontology in the form of a *UML class model*. Whenever the context is clear, we omit the name space prefix UML and simply say 'object', 'class', etc., instead of 'UML object', 'UML class', etc.

3.1   Classes and Objects

In the UML specification, *"an object represents a particular instance of a class. It has identity and attribute values."* While in the UML objects are instances of classes, individuals are instances of universals in GFO. A *"Class describes a set of Objects sharing a collection of Features, including Operations, Attributes and Methods, that are common to the set of Objects."* [p. 2-26] *"The model is concerned with describing the intension of the class, that is, the rules that define it. The run-time execution provides its extension, that is, its instances."* [p. 3-35]

We may observe a direct correspondence between universals and classes of a certain kind, as stated in the following principle:

Principle 1: In a *UML-ontology*, any universal U of the domain may be represented as a concrete class C_U. Conversely, for all concrete classes (of a *UML-ontology*) whose instances are basic objects or links (representing individuals), there must be a corresponding universal in the domain.

In a *UML-ontology*, any individual of the domain that is an instance of a universal may be represented as an object (or link) of the class representing the universal. For every universal *U* there is a set *Ext(U)*, called its extension, containing all instances of *U* as elements. Even if two universals U_1 and U_2 have identical extensions ($Ext(U_1) = Ext(U_2)$), they are considered as different universals. As a consequence, we can account for concomitant (co-extensional) universals, such as fluid and viscous and living and mortal but, which clearly have different intentions.

Most classes in a *UML-ontology* represent Substance Universals. This is due to the fact that Substantials are prior to Moments not only from an existential but also from an identification point of view. For example, Schneider [13] claims that moments (tropes) are *identificationally dependent* on substances (objects), i.e., while the latter

can be 'single out on their own', in order to identify a moment *m* of substantial *s*, one has to identify *s* first.

The substances (belonging to the realm of concrete reality) that we talk and think about can be classified in all kinds of ways. We can sort things by color (e.g. creating the class of red things) or by shape (e.g. the class of things with circular form) or by (clusters of) properties that define classes of things such as the classes of elephants, oak trees, cars, europeans and students. One important question that arises at this point is: if things can be classified in a multitude of ways how do we create the conceptual categories used in cognition and language? Can we provide methodological guidelines that assist an ontology designer in evaluating modeling alternatives?

The development of a theory of substance universals that addresses these questions, due to its importance and complexity, cannot be dealt in here and deserves a paper on its own. In [2], we present a philosophically and psychologically well-founded formal theory of substance universals for conceptual modeling which is used to propose: (a) a profile for UML whose elements represent finer-grained distinctions between different types of substance universals; (b) a set of constraints defining the admissible relations between these elements. The categories in this profile provide a foundation for a number of modeling primitives that, albeit often used, are commonly defined in an ad hoc manner in the practice of conceptual modeling and knowledge representation (e.g. kind, phase or state, role, mixin). In the remaining of this paper, non-stereotyped classes that appear in the models represent substance universals.

3.2 Attributes and Data Types

Suppose that we have an extension of the situation illustrated in figure 3, i.e. a substantial universal *Apple* whose elementary specification contains the features *Weight* and *Color*. Thus, for an instance *a* of Apple there are instances *c* of the quality universal color and *w* of weight both inhering in *a*. The intention of this universal could be represented by the following quality specification: $\forall a$ (a::Apple $\rightarrow \exists c \exists w$ (c::Color $\wedge i(c,a)) \wedge$ (w::Weight $\wedge i(w,a)))$.

Associated with *w* there is a quale *q* denoting a particular weight value, i.e. a point in the weight quality dimension such that *ql(q,w)* holds. We assume the weight quality domain to be a one-dimensional structures isomorphic to the half-line of nonnegative numbers represented by the set **WeightValue**. The mapping between a substance *a* and its weight quale can be represented by the following function *weight_in_grams*: **Ext(Apple) $\rightarrow$ WeightValue** such that *weight_in_grams* (x) = {y $\in$ **WeightValue** | $\exists z$::Weight $i(z,x) \wedge ql(y,z)$}.

In general, let U be a substance or quality universal and Q be a quality universal. Let E be a quality specification capturing the intention of universal U: $\forall x$ (x::U $\rightarrow \exists y$ (y::Q $\wedge i(c,a)))$. If D is a quality dimension associated with Q, we can define the function f:**Ext(U) $\rightarrow$ D** (named an *attribute function*) for S such that for every x::U we have that $f(x) = \{y \mid \exists z$::Q $i(z,x) \wedge ql(y,z)\}$. Analogous attribute functions can be defined for every quality universal Q_i in a quality specification of universal U'.

In the simplest case, quality universals appearing in the quality specification of *U* can be represented in a *UML-ontology* via their corresponding *attribute functions* and associated *quality dimensions* in the following manner:

Principle 2: Every attribute function associated to the elementary specification of the universal U may be represented as an attribute of the class C_U (representation of the universal U) in a *UML-ontology*; every quality dimension which is the co-domain of one these functions may be represented as data types of the corresponding attributes in this *UML-ontology*.

In UML, *"a data type is a special kind of classifier, similar to a class, whose instances are values (not objects)...A value does not have an identity, so two occurrences of the same value cannot be differentiated"* [5, p. 95]. A direct representation of Apple's elementary specification in UML according to principle 2 maps the attribute function **weight_in_grams:Ext(Apple)→WeightValue** to an attribute weight_in_grams with data type WeightValue in class Apple (figure 4(a) and 4(b)).

In order to model the relation between the quality c (color) and its quale, there are other issues to be considered. As previously mentioned, the quality dimension associated with the Color universal is a three-dimension splinter (fig. 2) composed of quality dimensions hue, saturation and brightness. In DOLCE, these dimensions are considered to be indirect qualities, i.e., there are quality individuals h, s, b instances of quality universals Hue, Saturation and Brightness, respectively, that inhere in the color quality c (which in turn inheres in substance a). For this reason h, s, b are named indirect qualities of a. The intention of the universal Color could then be represented by the following specification: **∀c (c::Color → ∃h∃s∃b (h::Hue ∧ i(h,c)) ∧(s::Saturation ∧ i(s,c)) ∧(b::Brightness ∧ i(b,c))).**

In this case, we can derive the following attribute functions from the features in this specification: (a) **hue: Ext(Color) → HueValue**; (b) **saturation: Ext(Color) → SaturationValue**; (c) **brightness: Ext(Color) → BrightnessValue**. Together these functions map each quality of a color *c* to its corresponding quality dimension. One possibility for modeling this situation is a direct application of principle 2 to the Color universal quality specification. In this alternative, depicted in figure 4(a), the class Color directly represents the quality universal color and, its attributes the attribute functions *hue*, *saturation* and *brightness*.

Another modeling alternative is to use the UML construct of a data type to represent a quality domain and its constituent quality dimensions (figure 4(b)). In this case, first we define for the universal Apple an attribute function *color***: Ext(Apple) → HueValue × SaturationValue × BrightnessValue** such that *color***(x) = {<y,z,w> ∈ HueValue × SaturationValue × BrightnessValue | ∃c::Color i(c,x) ∧ (y = hue(c)) ∧ (z = saturation(c)) ∧ (w = brightness(c))}.**

In figure 4(b), the data type fields *hue*, *staturation*, *brightness* do not represent attribute functions but values (qualia) that form the data type tuple so that the "instances"(members) of ColorValue are tuples ⟨x,y,z⟩ where x ∈ HueValue, y ∈ SaturationValue and z ∈ BrightnessValue (and it is unfortunate, in this sense, that UML uses the same notation for both). The *navigable end name color* in the association between Apple and ColorValue represents the attribute function *color* described above. *Navigable end names* are a suitable alternative mechanism for representing attribute functions since in UML they are semantically equivalent to attributes [5,p. 82].

One should notice that these two forms of representation do not convey the same information (a fact which we highlight by the use of different stereotypes): in figure

4(a), color objects are one-sidedly existentially dependent on the individuals they are related to via the *inheres in* relation. These objects are *bonafide* individuals with a definite numerical identity. In figure 4(b), contrariwise, the members of the Color-Value are *pure values* that represent points in a quality domain. These values can qualify a number of different objects but they exist independently of them in the sense that a color tuple is a part of quality domain even if no object "has that color". Both representations are warranted, in the sense that ontologically consistent interpretations can be found in both cases and, which alternative shall be pragmatically more suitable is a matter of empirical investigation.
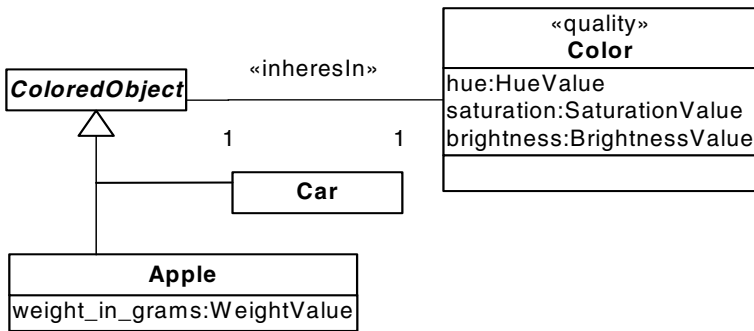
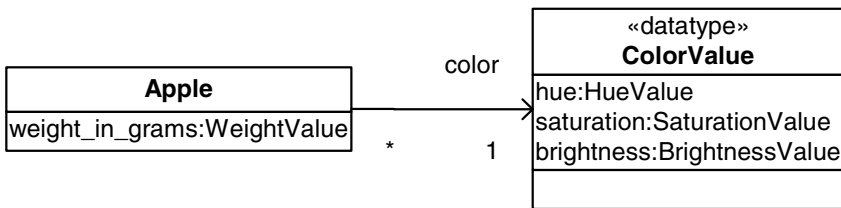**Fig. 4(a).** Representing Quality Universals and indirect qualities

Fig. 4(b). Representing Qualia in a multi-dimensional quality domain

Notwithstanding, we believe that some guidelines could be anticipated. In situations in which the qualities of a quality all take their values (qualia) in a single quality domain (e.g. color), the latter alternative should be preferred due to its compatibility with the modeling tradition in conceptual modeling and knowledge representation. However, there are cases in which we want to directly represent the quality associated with a substance, not its qualia. An example of such a situation is depicted in Figure 5, which models the relation between a Hospital, its Patients, and a number of symptoms reported by these patients. Suppose an individual patient John is suffering from headache and influenza. John's headache and influenza are qualities inhering in John. Even

if another patient, for example Paul, has a headache that is qualitatively indistinguishable from that of his, John's headache and Paul's headache are two different individuals. Moreover, instances of Symptoms can have qualities themselves (such as duration and intensity) and can participate in relations of, for example, causation or precedence. In figure 6, the quality universal Symptom is represented by a class construct decorated with the «quality» stereotype. The formal relation between Symptom and Patient is mapped to the *inherence* relation in the instance level, representing the existential dependence of a Symptom on a Patient. In other words, for an instance *s* of Symptom there must be a specific instance *p* of Patient associated with *s*, and in every situation that *s* exists *p* must exist and the inherence relation between the two must hold. One should notice that this formal relation has a semantics which is outside the usual interpretation of the association construct in UML. According to its standard usage, the multiplicity 1 in the Patient end only demands that, in every situation, symptom *s* must be related to *an* instance of Patient. The inherence relation, however, requires *s* to be always related to the *one and the same instance* of Patient. The difference between these two sorts of requirements is analogous to those marking the difference between essential and mandatory part-whole relations [1].

Finally, quality domains are composed of integral dimensions, which means that the value of one dimension cannot be represented without representing the values of others. By representing the color quality domain in terms of a quality universal (or data type) we can reinforce (via its constructor method) that its tuples will always have values for all the integral dimensions. Additionally, the representation of a quality domain should account not only for its quality dimensions but also for the constraints on the relation between them imposed by its structure. To mention another example, consider the Gregorian calendar as a quality domain (composed of the linear quality dimensions days, months and years) in which date qualities can be represented. It is clear that the value of one dimension constrains the value of the others in a way that, for example, the points [31-April-2004] and [29-February-2003] do not belong to this quality structure. Once more constraints represented on the constructor method of a «quality» class (or data type) can be used to restrict the possible tuples that can be instantiated.
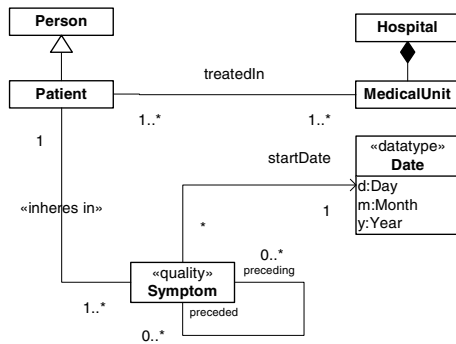

Fig. 5. Representing Quality Universals and Formal Relational Universals

In the sequel, we observe the following principle between quality domains and their representation in terms of data types:

Principle 3: Every quality dimension D associated to a quality universal Q may be represented as a data type DT in a *UML-ontology*; A set of integral dimension D_1… D_n (represented by data types DT_1… DT_n) constituting a quality domain QD can be grouped in data type W representing QD. In this case, every quality dimension D_i of QD may be represented by a field of W of type DT_i. Moreover, the relations between the dimensions D_i of QD may be represented by constraints relating the attributes of data type W.

3.3   Associations

In the UML, the ER concept of a relationship type is called association. "An association defines a semantic relationship between classifiers. *The instances of an association are a set of tuples relating instances of the classifiers.… An instance of an association is a link, which is a tuple of instances drawn from the corresponding classifiers*". The OMG UML Specification is somehow ambiguous in defining associations. An association is primarily considered to be a 'connection', but, in certain cases (whenever it has 'class-like properties'), an association may be a class: "An association class is an association that is also a class. It not only connects a set of classifiers but also defines a set of features that belong to the relationship itself and not any of the classifiers."

An association A between the classes $C_1,…,C_n$ of a UML-ontology can be understood in our framework as a relation (relational universal) *R* between the corresponding universals $U_1,…,U_n$ whose extension consists of all tuples corresponding to the links of A. In figure 5, an example of a formal relation is *precedence*. Precedence is a partial order relation between symptoms that depends only on the starting moment of each of them. The relation *treatedIn* between *Patient* and *MedicalUni,* contrariwise, requires the existence of a third entity, namely a *Treatment* process, in order for the relation to hold. This latter case can be modeled in our framework as follows: Let *treatedIn* be a binary material association corresponding to a relator universal *Treatment* whose instances are individual treatment processes. These individual *treatment* processes connect two individuals: a *patient*, say *John*, and a *MedicalUnit*, say *TraumaUnit#1*. Thus, **[John, TraumaUnit#1]:$R_{treatedIn}$(Person, MedicalUnit).** Since *John::Person* and *TraumaUnit#1::MedicalUnit*, and there is a specific treatment process *t::Treatment* we have the fact: **⟨t:John,TraumaUnit#1⟩.**

We obtain the definition for the tuple *[a_1,a_2]* being a link of the association *treatedIn* between *Person* and *MedicalUnit*: **[a_1,a_2]:$R_{treatedIn}$(Person, MedicalUnit) ↔ a_1::Person ∧ a_2::MedicalUnit ∧ ∃t(t::Treatment ∧ ⟨t:a_1a_2⟩::F(Treatment)).**

We can now state the following principles regarding the representation of formal and material relations in a *UML-ontology*:

Principle 4: In a *UML-ontology*, any formal relation universal R_F of the domain may be represented as a standard association whose links represent the tuples in the extension of R_F. Conversely, a material relation R_M of the domain may be represented in a

UML-ontology by representing the *relator universal* associated with the relation as an association class.

There is a specific practical problem concerning the representation of *material relations* as standard associations that supports the modeling choice represented in this principle. This problem, pointed by Bock and Odell in [14], is caused by the fact that the standard notation collapses two different types of *multiplicity constraints*. Let us take, for instance, the association *treatedIn* depicted in figure 6. The model states that each Patient can be treated in one-to-many Medical Units and that each medical unit can treat one-to-many patients. However, this statement is ambiguous since many different interpretations can be given to it, including the following: (i) a patient is related to only one treatment to which participate possibly several medical units; (ii) a patient can be related to several treatments to which only one single unit participates; (iii) a patient can be related to several treatments to which possibly several medical units participate; (iv) several patients can be related to a treatment to which several medical units participate, and a single patient can be related to several treatments. The cardinality constraint that indicates how many patients (or medical units) can be related to one instance of Treatment is named *single-tuple* cardinality constraints. *Multiple-tuple* cardinality constraints restrict the number of treatments a patient (or medical unit) can be related to. By modeling the relator universal *Treatment* as an association class one can explicit represent both types of constraints. A version of figure 5 adopting this principle is presented in figure 6.

This problem is specific to material relations. Extensional formal relations are sets of tuples, i.e. an instance of the relation is itself a tuple with predefined arity. In formal relations, cardinality constraints are always unambiguously interpreted as being *multiple-tuple* (since there is no point in specifying single-tuple cardinality constraints for a relation with predefined arity). Hence, formal relations can be suitably represented as standard UML associations. One should notice that the relations between Patient and Treatment, and Medical Unit and Treatment are formal relations between universals (*inheres in*). This is important to block the infinite regress that arises if material relations are required to relate these entities.
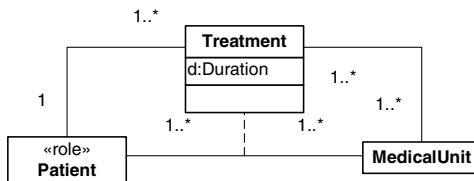
Fig. 6. Representing Material Relational Universals

In the same way as qualities, relators can have their own inhering moments (e.g. Duration, as a quality associated to the universal Treatment, in fig. 6) or they can be used as a foundation for other relations such as, for instance, a relator universal *Invoice* whose instances connect particular Treatments and Payers.

4   Final Considerations

The development of a well-grounded, axiomatized upper level ontology is an important step towards the definition of real-world semantics for ontology representation diagrammatic languages. In this paper, we use the General Formalized Ontology underlying the language GOL to evaluate the ontological correctness of UML as an ontology representation language, and to develop guidelines that assign well-defined ontological semantics to UML constructs. In particular, we focus on the ontology representation most basic primitives: class, attribute, data types and association.

However, despite the importance of these modeling constructs, there is still a deficiency of methodological support for helping the user of the language deciding how to model the elements of a given domain. For example, as reported in [15], the same real-world phenomena (e.g. Marriage) can sometimes be modeled as a class, a relation or an attribute. This situation is made worse by the fact that there is not in the literature a shared agreement on the ontological meaning of these constructs. To cite one example: in a series of papers (e.g. [15,16]) the proponents of the BWW (Bunge-Wand-Weber) approach claim that universals whose instances are properties (moments) should not be modeled as classes in a conceptual model of the domain. This claim is contested by Veres and Hitchman in [17] who employ Jakendoff's Semantic Structures [18] as well as empirical results from modeling sections with practitioneers. In this paper, by presenting a principled interpretation for what an association is supposed to denote, we show that representing relational moments as classes is not only ontologically correct but also beneficial from a practical point of view.

In sum, the work presented here is part of larger effort that aims at developing: (i) a formal ontological framework that can be used as a system of domain-independent meta-level categories to provide ontological semantics for ontology representation languages [3,4]; (ii) a set of well-founded methodological tools (language extensions, guidelines, profiles and design patterns) that contribute to the discipline of ontological engineering [1,2].

References

1. Guizzardi, G., Herre, H., Wagner G.: Towards Ontological Foundations for UML Conceptual Models. 1st International Conference on Ontologies Databases and Applications of Semantics (ODBASE), USA, 2002.
2. Guizzardi, G., Wagner G., Guarino, N., van Sinderen, M. An Ontologically well-Founded Profile for UML Conceptual Models, 16th International Conference on Advanced Information Systems Engineering (CAiSE), Latvia, 2004.
3. Degen, W., Heller B., Herre H., Smith, B.: GOL: Towards an axiomatized upper level ontology. 2nd International Conference of Formal Ontologies and Information Systems (FOIS'01), USA, 2001.
4. Heller, B., Herre, H. Ontological Categories in GOL. Axiomathes 14: 71-90 Kluwer Academic Publishers, 2004.
5. Object Management Group, UML 2.0 Superstructure Specification, Doc.# ptc/03-08-02, Aug. 2003.

6. Cranefield, S., Purvis M.: UML as an ontology modelling language, Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99), Germany, 1999.
7. Object Management Group, Ontology Definition Metamodel Request for Proposals, OMG Document: ad/2003-03-40, 2003.
8. Loebe, F. An Analysis of Roles: Towards Ontology-Based Modelling, Diploma Thesis, Institute for Medical Informatics, Statistics and Epidemiology (IMISE), University of Leipzig, 2003.
9. Gärdenfors, P. Conceptual Spaces: the Geometry of Thought. MIT Press, USA, 2000.
10. Masolo, C.; Borgo, S.; Gangemi, A.; Guarino, N.; Oltramari, A.; Ontology Library, WonderWeb Deliverable D18, 2003.
11. Smith, B., Fiat Objects, 11th European Conference on Artificial Intelligence, The Netherlands, 1994.
12. Smith, B.; Mulligan, K.; A Relational Theory of the Act, Topoi (5/2), 115-30,1986.
13. Schneider, L., Designing Foundational Ontologies: The Object-Centered High-Level Reference Ontology OCHRE as a Case Study, 22th International Conference on Conceptual Modeling (ER), USA, 2003.
14. Bock, C.; Odell, J., A More Complete Model of Relations and Their Implementation Part I: Relations as Object Types, Journal Of Object-Oriented Programming Vol 10, No 3. June 1997.
15. Wand. Y.; Storey, V.C.; Weber, R.; An Ontological Analysis of the Relationship Construct in Conceptual Modeling, ACM Transactions on Database Systems, Vol.24, No.4, Dec.1999.
16. Evermann, J. and Wand, Y. Towards ontologically based semantics for UML constructs. In H. Kunii, S. Jajodia, and A. Solvberg, editors, Proceedings of the 20th International Conference on Conceptual Modeling (ER), Japan, 2001.
17. Veres, C.; Hitchman, S, Using Psychology to Understand Conceptual Modeling, 10th European Conference on Information Systems (ECIS 2002), Poland.
18. Jakendoff, R.S. Semantic Structures, Current Studies in Linguistics, MIT Press, 1990.