Published in, Laender, A., and Storey, V., eds., *Proceedings of ER-2000: The International Conference on Conceptual Modeling.* October, 2000. Springer-Verlag LNCS.

Ontological Analysis of Taxonomic Relationships

Nicola Guarino and Christopher Welty†
LADSEB/CNR
Padova, Italy
{guarino,welty}@ladseb.pd.cnr.it
http://www.ladseb.pd.cnr.it/infor/ontology/ontology.html
† on sabbatical from Vassar College, Poughkeepsie, NY

Abstract. Taxonomies based on a partial-ordering relation commonly known as is-a, class inclusion or subsumption have become an important tool in conceptual modeling. A well-formed taxonomy has significant implications for understanding, reuse, and integration, however the intuitive simplicity of taxonomic relations has led to widespread misuse, making clear the need for rigorous analysis techniques. Where previous work has focused largely on the semantics of the *is-a* relation itself, we concentrate here on the ontological nature of the *arguments* of this relation, in order to be able to tell whether a single *is-a* link is ontologically well-founded. For this purpose, we discuss techniques based on the philosophical notions of *identity, unity, essence,* and *dependence*, which have been adapted to the needs of information systems design. We demonstrate the effectiveness of these techniques by taking real examples of poorly structured taxonomies, and revealing cases of invalid generalization. The result of the analysis is a cleaner taxonomy that clarifies the modeler's ontological commitments.

1 Introduction

Taxonomies are an important tool in conceptual modeling, and this has been especially true since the introduction of the extended ER model [6,26]. Properly structured taxonomies help bring substantial order to elements of a model, are particularly useful in presenting limited views of a model for human interpretation, and play a critical role in reuse and integration tasks. Improperly structured taxonomies have the opposite effect, making models confusing and difficult to reuse or integrate.

Many previous efforts at providing some clarity in organizing taxonomies have focused on the semantics of the taxonomic relationship (also called is-a, class inclusion, subsumption, etc.) [3], on different kinds of relations (generalization, specialization, subset hierarchy) according to the constraints involved in multiple taxonomic relationships (covering, partition, etc.) [23], on the taxonomic relationship in the more general framework of data abstractions [7], or on structural similarities between descriptions [2,5]. Our approach differs in that we focus on the arguments (i.e. the properties) involved in the taxonomic relationship, rather than on the semantics of the relationship itself. The latter is taken for granted, as we take the statement "φ subsumes ϕ" to mean simply:

$$\forall x \; \phi(x) \rightarrow \varphi(x) \tag{1}$$

Our focus here will be on verifying the plausibility and the well-foundedness of single statements like (1) on the basis of the *ontological nature* of the two properties ϕ and φ.

In this paper we present and formalize four fundamental notions of so-called *Formal Ontology* [11]: *identity, unity, essence, and dependence*, and then show how they can be used as the foundation of a methodology for conceptual modeling. Implicitly, we also assume a fifth fundamental notion, *parthood*, whose role for conceptual analysis has been extensively discussed elsewhere [1,22]. Finally, we demonstrate the effectiveness of our methodology by going through a real example of a poorly structured taxonomy, and revealing cases of invalid generalization. The result of the analysis is a cleaner taxonomy that clarifies the modeler's ontological assumptions.

2 Background

The notions upon which our methodology is based are subtle, and before describing them with formal rigor we discuss the basic intuitions behind them and how they are related to some existing notions in conceptual modeling.

2.1 Basic Notions

Before presenting our formal framework let us informally introduce the most important philosophical notions: *identity*, *unity*, *essence*, and *dependence*. The notion of identity adopted here is based on intuitions about how we, as cognitive agents, in general interact with (and in particular recognize) individual entities in the world around us. Despite its fundamental importance in Philosophy, it has been slow in making its way into the practice of conceptual modeling for information systems, where the goals of analyzing and describing the world are ostensibly the same.

The first step in understanding the intuitions behind identity requires considering the distinctions and similarities between *identity* and *unity*. These notions are different, albeit closely related and often confused under a generic notion of identity. Strictly speaking, identity is related to the problem of distinguishing a specific instance of a certain class from other instances of that class by means of a *characteristic property*, which is unique for *it* (that *whole* instance). Unity, on the other hand, is related to the problem of distinguishing the *parts* of an instance from the rest of the world by means of a *unifying relation* that binds the parts together, and nothing else. For example, asking, "Is that my dog?" would be a problem of identity, whereas asking, "Is the collar part of my dog?" would be a problem of unity.

Both notions encounter problems when time is involved. The classical one is that of *identity through change*: in order to account for common sense, we need to admit that an individual may remain *the same* while exhibiting different properties at different times. But which properties can change, and which must not? And how can we reidentify an instance of a certain property after some time? The former issue leads to the notion of an *essential property*, on which we base the definition of *rigidity*, discussed below, while the latter is related to the distinction between *synchronic* and *diachronic* identity. An extensive analysis of these issues in the context of conceptual modeling has been made elsewhere [14].

The fourth notion, *ontological dependence*, may involve many different relations such as those existing between persons and their parents, holes in pieces of cheese and the cheese, and so on [22]. We focus here on a notion of dependence as applied to properties. We distinguish between *extrinsic* and *intrinsic* properties, according to whether they depend or not on other objects besides their own instances. An intrinsic

property is typically something inherent to an individual, not dependent on other individuals, such as having a heart or having a fingerprint. Extrinsic properties are not inherent, and they have a relational nature, like "being a friend of John". Among these, there are some that are typically assigned by external agents or agencies, such as having a specific social security number, having a specific customer i.d., even having a specific name.

It is important to note that our ontological assumptions related to these notions ultimately depend on our *conceptualization* of the world [12]. This means that, while we shall use examples to clarify the notions central to our analysis, *the examples themselves will not be the point of this paper.* For example, the decision as to whether a cat remains the same cat after it loses its tail, or whether a statue is identical with the marble it is made of, are ultimately the result of our sensory system, our culture, etc. The aim of the present analysis is to clarify the formal tools that can both make such assumptions explicit, and reveal the logical consequences of them. When we say, e.g. that "having the same fingerprint" may be considered an identity criterion for *PERSON*, we do *not* mean to claim this is the universal identity criterion for *PERSON*s, but that *if this were* to be taken as an identity criterion in some conceptualization, what would that mean for the property, for its instances, and its relationships to other properties?

2.2 Related Notions

Identity has many analogies in conceptual modeling for databases, knowledge bases, object-oriented, and classical information systems, however none of them completely captures the notion we present here. We discuss some of these cases below.

Membership conditions. In description logics, conceptual models usually focus on the sufficient and necessary criteria for class *membership*, that is, recognizing instances of certain classes [4]. This is not identity, however, as it does not describe how instances of the same class are to be told apart. This is a common confusion that is important to keep clear: membership conditions determine when an entity is an instance of a class, i.e. they can be used to answer the question, "Is that *a* dog?" but not, "Is that *my* dog?"

Globally Unique IDs. In object-oriented systems, uniquely identifying an object (as a collection of data) is critical, in particular when data is persistent or can be distributed [28]. In databases, *globally unique id's* have been introduced into most commercial systems to address this issue. These solutions provide a notion of identity for the descriptions, for the units of data (objects or records), but not for the entities they describe. It still leaves open the possibility that two (or more) descriptions may refer to the same *entity*, and it is this entity that our notion of identity is concerned with. In other words, globally unique IDs can be used to answer, "Is this the same description of a dog?" but not, "Is this my dog."

Primary Keys. Some object-oriented languages provide a facility for overloading or locally defining the equality predicate for a class. In standard database analysis, introducing new tables requires finding unique keys either as single fields or combinations of fields in a record. These two similar notions very closely approach our notion of identity as they do offer evidence towards determining when two descriptions refer to the same entity. There is a very subtle difference, however, which we will attempt to

briefly describe here and which should become more clear with the examples at the end of the paper.

Primary (and candidate) keys and overloaded equality operators are typically based on *extrinsic properties* (see Section 2.1) that are required by a system to be unique. In many cases, information systems designers add these extrinsic properties simply as an escape from solving (often very difficult) identity problems. Our notion of identity is based mainly on *intrinsic properties*—we are interested in analyzing the inherent nature of entities and believe this is important for understanding a domain.

This is not to say that the former type of analysis never uses intrinsic properties, nor that the latter never uses extrinsic ones – it is merely a question of emphasis. Furthermore, our analysis is often based on information which *may not be represented in the implemented system*, whereas the primary key notion can never use such information. For example, we may claim as part of our analysis that people are uniquely identified by their brain, but this information would not appear in the final system we are designing. Our notion of identity and the notion of primary keys are not incompatible, nor are they disjoint, and in practice conceptual modelers will often need both.

3 The Formal Tools of Ontological Analysis

In this section we shall present a formal analysis of the basic notions discussed above, and we shall introduce a set of *meta-properties* that represent the behaviour of a property with respect to these notions. Our goal is to show how these meta-properties impose some constraints on the way subsumption is used to model a domain.

Our analysis relies on certain fairly standard conventions and notations in logic and modal logic, which are described in more detail in [16]. We shall denote primitive meta-properties by bold letters preceded by the sign "+", "-" or "~" which will be described for each meta-property. We use the notation $\phi^{\mathbf{M}}$ to indicate that the property ϕ has the meta-property **M**.

3.1 Rigidity

The notion of rigidity was defined previously in [10] as follows:

Definition 1 A r*igid property* is a property that is essential to *all* its instances, i.e. a property ϕ such that: $\forall x\, \phi(x) \rightarrow \Box \phi(x)$.

Definition 2 A *non-rigid property* is a property that is not essential to *some* of its instances, i.e. $\exists x\, \phi(x) \wedge \neg\Box \phi(x)$.

Definition 3 An *anti-rigid property* is a property that is not essential to *all* its instances, i.e. $\forall x\, \phi(x) \rightarrow \neg\Box \phi(x)$.

For example, we normally think of *PERSON* as rigid; if *x* is an instance of *PERSON*, it must be an instance of *PERSON* in every possible world. The *STUDENT* property, on the other hand, is normally not rigid; we can easily imagine an entity moving in and out of the *STUDENT* property while being the same individual.

Anti-rigidity was added as a further restriction to non-rigidity. The former constrains all instances of a property and the latter, as the simple negation of rigidity, constrains at least one instance. Anti-rigidity attempts to capture the intuition that all instances of certain properties must possibly not be instances of that property. Consider the prop-

erty *STUDENT*, for example: in its normal usage, every instance of *STUDENT* is not necessarily so.

Rigid properties are marked with the meta-property **+R**, non-rigid properties are marked with **-R**, and anti-rigid properties with **~R**. Note that rigidity as a meta-property is not "inherited" by sub-properties of properties that carry it, e.g. if we have $PERSON^{+\mathbf{R}}$ and $\forall x\, STUDENT(x) \rightarrow PERSON(x)$ then we know that all instances of *STUDENT* are necessarily instances of *PERSON*, but not *necessarily* (in the modal sense) instances of *STUDENT*, and we furthermore may assert $STUDENT^{\sim\mathbf{R}}$. In simpler terms, an instance of *STUDENT* can cease to be a student but may not cease to be a person.

3.2 Identity

In the philosophical literature, an *identity condition* (IC) for an arbitrary property ϕ is usually defined as a suitable relation ρ satisfying the following formula:

$$\phi(x) \wedge \phi(y) \rightarrow (\rho(x, y) \leftrightarrow x = y) \tag{2}$$

For example, the property *PERSON* can be seen as carrying an IC if relations like *having-the-same-SSN* or *having-the-same-fingerprints* are assumed to satisfy (2).

As discussed in more detail elsewhere [14], the above formulation has some problems, in our opinion. The first problem is related to the need for distinguishing between *supplying* an IC and simply *carrying* an IC: it seems that non-rigid properties like *STUDENT* can only carry their ICs, inheriting those supplied by their subsuming rigid properties like *PERSON*. The intuition behind this is that, since the same person can be a student at different times in different schools, an IC allegedly supplied by *STUDENT* (say, having the same registration number) may be only local, within a certain studenthood experience.

The second problem regards the nature of the ρ relation: what makes it an IC, and how can we index it with respect to time to account for the difference between *synchronic* and *diachronic* identity?

Finally, deciding whether a property carries an IC may be difficult, since finding a ρ that is both necessary *and* sufficient for identity is often hard, especially for natural kinds and artifacts.

For these reasons, we have refined (2) as follows:

Definition 4 An *identity condition* is a formula Γ that satisfies either (3) or (4) below, excluding trivial cases and assuming a predicate E for *actual existence* (see [15]):

$$\mathrm{E}(x,t) \wedge \phi(x,t) \wedge \mathrm{E}(y,t') \wedge \phi(y,t') \wedge x{=}y \rightarrow \Gamma(x,y,t,t') \tag{3}$$

$$\mathrm{E}(x,t) \wedge \phi(x,t) \wedge \mathrm{E}(y,t') \wedge \phi(y,t') \wedge \Gamma(x,y,t,t') \rightarrow x{=}y \tag{4}$$

An IC is necessary if it satisfies (3) and sufficient if it satisfies (4). Based on this, we define two meta-properties:

Definition 5 Any property *carries* an IC iff it is subsumed by a property supplying that IC (including the case where it supplies the IC itself).

Definition 6 A property ϕ *supplies* an IC iff i) it is rigid; ii) there is a necessary or sufficient IC for it; and iii) The same IC is not carried by *all* the properties subsuming ϕ. This means that, if ϕ inherits different (but compatible) ICs from multiple properties, it still counts as supplying an IC.

Definition 7 Any property carrying an IC is called a *sortal* [25].

Any property carrying an IC is marked with the meta-property **+I** (**-I** otherwise). Any property supplying an IC is marked with the meta-property **+O** (**-O** otherwise). The letter "O" is a mnemonic for "own identity". From the above definitions, it is obvious that **+O** implies **+I** and **+R**. For example, both *PERSON* and *STUDENT* do carry identity (they are therefore **+I**), but only the former *supplies* it (**+O**).

3.3 Unity

In previous work we have extensively discussed and formalized the notion of unity, which is itself based upon the notion of part [14]. This formalization is based on the intuition that a whole is something all of whose parts are connected in such a way that each part of the whole is connected to all the other parts of that whole and nothing else. Briefly, we define:

Definition 8 An object *x is a whole under* ω iff ω is an equivalence relation such that all the parts of *x* are linked by ω, and nothing else is linked by ω.

Definition 9 A property ϕ *carries a unity condition* iff there exists a single equivalence relation ω such that each instance of ϕ is a whole under ω.

Depending on the ontological nature of the ω relation, which can be understood as a "generalized connection", we may distinguish three main kinds of unity for concrete entities (i.e., those having a spatio-temporal location). Briefly, these are:

- *Topological unity*: based on some kind of topological or physical connection, such as the relationship between the parts of a piece of coal or an apple.
- *Morphological unity*: based on some combination of topological unity and shape, such as a ball, or a morphological relation *between wholes* such as for a constellation.
- *Functional unity*: based on a combination of other kinds of unity with some notion of purpose as with artifacts such as hammers, or a functional relation between wholes as with artifacts such as a bikini.

As the examples show, nothing prevents a whole from having parts that are themselves wholes (with a different UC). This can be the foundation of a theory of *pluralities*, which is however out of this paper's scope.

As with rigidity, in some situations it may be important to distinguish properties that do not carry a *common* UC for all their instances, from properties all of whose instances are not wholes. As we shall see, an example of the former kind may be *LEGAL AGENT*, all of whose instances are wholes, although with different UCs (some legal agents may be people, some companies). *AMOUNT OF MATTER* is usually an example of the latter kind, since none of its instances can be wholes. Therefore we define:

Definition 10 A property has *anti-unity* if every instance of the property is not a whole.

Any property carrying a UC is marked with the meta-property **+U** (**-U** otherwise). Any property that has anti-unity is marked with the meta-property **~U**, and of course **~U** implies **-U**.

3.4 Dependence

The final meta-property we employ as a formal ontological tool is based on the notion of dependence. As mentioned in Section 2.1, we focus here on ontological dependence

as applied to properties. The formalization adopted below is refined from previous work [9], and is based on Simons' definition of *notional dependence* [22]. We are aware that this is only an approximation of the more general notion of extrinsic (or relational) property, and that further work may be needed (see for instance [19]).

Definition 11 A property ϕ is *externally dependent* on a property ψ if, for all its instances x, necessarily some instance of ψ must exist, which is not a part nor a constituent of x:

$$\forall x \, \Box (\phi(x) \rightarrow \exists y \, \psi(y) \wedge \neg P(y, x) \wedge \neg C(y, x)) \quad (5)$$

The part and constituent relations are discussed further in [16]. In addition to excluding parts and constituents, a more rigorous definition must exclude qualities (such as colors), things which necessarily exist (such as the universe), and cases where ψ is subsumed by ϕ (since this would make ϕ dependent on itself). Intuitively, we say that, for example, *PARENT* is externally dependent on *CHILD* (one can not be a parent without having a child), but *PERSON* is not externally dependent on heart nor on body (because any person has a heart as a part and is constituted of a body).

An externally dependent property is marked with the meta-property **+D** (**-D** otherwise).

3.5 Constraints and Assumptions

Our meta-properties impose several constraints on taxonomic relationships, and to these we add several methodological points that help to reveal modeling problems in taxonomies.

A first observation descending immediately from our definitions regards some *subsumption constraints.* If ϕ and ψ are two properties then the following constraints hold:

$\phi^{\sim \mathbf{R}}$ can't subsume $\psi^{+\mathbf{R}}$ (6)

$\phi^{+\mathbf{I}}$ can't subsume $\psi^{-\mathbf{I}}$ (7)

$\phi^{+\mathbf{U}}$ can't subsume $\psi^{-\mathbf{U}}$ (8)

$\phi^{\sim \mathbf{U}}$ can't subsume $\psi^{+\mathbf{U}}$ (9)

$\phi^{+\mathbf{D}}$ can't subsume $\psi^{-\mathbf{D}}$ (10)

Properties with incompatible ICs/UCs are disjoint. (11)

Constraints (6-10) follow directly from our meta-property definitions (see [15] for more discussion and examples), and (11) should be obvious from the above discussion of identity and unity, but it is largely overlooked in many practical cases [13,15]. Concrete examples will be discussed at the end of this paper.

Finally, we make the following assumptions regarding identity (adapted from [20]):

- *Sortal Individuation.* Every domain element must instantiate some property carrying an IC (**+I**). In this way we satisfy Quine's dicto "No entity without identity" [21].
- *Sortal Expandability.* If two entities (instances of different properties) are the same, they must be instances of a property carrying a condition for their identity.

4 Methodology

We are developing a methodology for conceptual analysis whose specific goal is to *make modeling assumptions clear*, and to produce *well-founded taxonomies.* The anal-

ysis tools that make up our methodology can be grouped into four distinct layers, such that the notions and techniques within each layer are based on the notions and techniques in the layers below.

4.1 First Layer: Foundations

In the lowest, foundational, layer of the methodology are the meta-properties described in Section 3, and in more detail in [16].

4.2 Second Layer: Useful Property Kinds

The second layer in the methodology contains an ontology of useful property kinds. This is an extension of the formal ontology of *basic* property kinds presented in [15], which includes further specializations of *sortal* properties (Def. 7), each one corresponding to an identity or unity condition commonly found in practice. This ontology can be seen as a library of reference cases useful to characterize the meta-properties of a given property, and to check for constraints violations. We sketch below the basic property kinds as well as some further specializations of sortal properties.

Basic Property Kinds. The formal ontology of properties discussed in [15] distinguishes eight different kinds of properties based on the valid and most useful combinations of the meta-properties discussed in Section 3. In this paper we mention only three of these combinations: *categories* (**+R-I**), *types* (**+R+O**), and *quasi-types* (**+R-O+I**). These and the other five property kinds add to a modeler's ability to specify the meaning of properties in an ontology, since the definition of each property kind includes an intuitive and domain-independent description of what part that kind of property should play in an ontology.

CO. Countable Properties. This is an important specialization of sortals. In many cases, besides carrying identity (**+I**), countable properties also carry unity (**+U**) . All subsumed properties must also be countable. Note that we appeal to a strict defintion of countability provided in [14], which may not be immediately intuitive in the case of collections, such as a group of people. One can count possible groups of people in a combinatoric sense, but by our current definition of countability, a group of people is not countable because it does not have unity.

ME. Properties carrying a mereologically extensional IC. Certain properties concerning masses or plural entities, such as *LUMP-OF-CLAY* or *GROUP-OF-PEOPLE*, have as a necessary identity condition that the parts of their instances must be the same (instances cannot change their parts). They cannot subsume properties with **-ME**.

UT. Properties carrying topological unity. See Section 3.3. Properties with **+UT** have unity (**+U**), and can not subsume properties with **-UT**.

UM. Properties carrying morphological unity. See Section 3.3. Properties with **+UM** have unity (**+U**), and can not subsume properties with **-UM**.

UF. Properties carrying functional unity. See Section 3.3. Properties with **+UF** have unity (**+U**), and can not subsume properties with **-UF**.

4.3 Third Layer: Ontology-Based Modeling Principles

The third layer in the methodology contains the notions of *backbone property* and *stratification*.

The backbone taxonomy. One of the principal roles of taxonomies is to impart structure on an ontology, to facilitate human understanding, and to enable integration. We have found that a natural result of our analysis is the identification of special properties in a taxonomy that best fill this role. We call these properties *backbone properties*, which constitute the *backbone taxonomy* [15].

The backbone taxonomy consists only of rigid properties, which are divided into three kinds (as discussed above): *categories*, *types*, and *quasi-types*. Categories can not be subsumed by any other kinds of properties, and therefore represent the highest level (most general) properties in a taxonomy. They are usually taken as primitive properties because defining them is too difficult (e.g. entity or thing).

Types are critical in our analysis because according to the assumptions presented in Section 3.5, *every instance instantiates at least one of these properties*. Therefore considering only the elements of the backbone gives someone a survey of the entire universe of possible instances.

Stratification. A very important result of our analysis is the recognition of multiple entities, based on different identity or unity criteria, where usually only one entity is conceived. The classical example is the statue and the clay it is made of, which count as different objects in our analysis. As discussed further in [13], this view results in a *stratified ontology*, where entities belong to different levels, depending on their identity and unity assumptions: we may distinguish for instance the physical level, the functional level, the intentional level, the social level. Entities at the higher levels are *constituted* (and co-located with) entities at the lower levels. The advantage of this view is a better semantic account of the taxonomic relation, a better account of the hidden ontological assumptions, and in general better ontologies. The costs are: i) a moderate proliferation (by a constant factor corresponding to the number of levels) of the number of entities in the domain; ii) the necessity to take into account different relations besides *is-a*, such as dependence, spatio-temporal colocalization, and constitution.

4.4 Fourth Layer: Top Level Ontology

The highest layer of our methodology is a top-level ontology designed using the notions and techniques of the layers below. This layer of the methodology is not yet complete, however first steps towards this have been discussed in [13].

4.5 Question/answer system

Finally, we are capturing the notions and techniques from these four layers in a knowledge-based question/answer system that guides conceptual modelers through the analysis process. This approach is similar to that of [24], and seemed necessary for two principal reasons:

- While rigidity and dependence tend to be simpler concepts to grasp, identity and unity are not. In addition to determining if a property carries identity and unity conditions, it is also useful to know, when possible, what those criteria are, because they must be consistent with subsuming and subsumed properties.

- Most of our analysis tools impose fairly strict constraints on the taxonomic links between properties, but verifying that these constraints are not violated can be tedious and difficult in human hands.

The Q/A system is implemented in CLASSIC [4], a description logic that provides simple rules and can perform subsumption testing between descriptions. Each meta-property, for example, is a concept in CLASSIC and properties in an ontology are individuals which are either derived or asserted to be instances of the concepts representing the meta-properties. All the principles, meta-properties, and constraints described in this paper have been represented in this system.

A demo of the system and a more detailed description is available on the web [27].

5 Example

We now discuss a more in-depth example. Figure 1 shows a messy taxonomy, which has mostly been drawn from existing ontologies such as WordNet, Pangloss, and Mikrokosmos. An initial scan of the taxonomy looks reasonable. To save space, we concentrate on just a few taxonomic pairs to explore how the system works. A more detailed discussion of the same example can be found in [16].

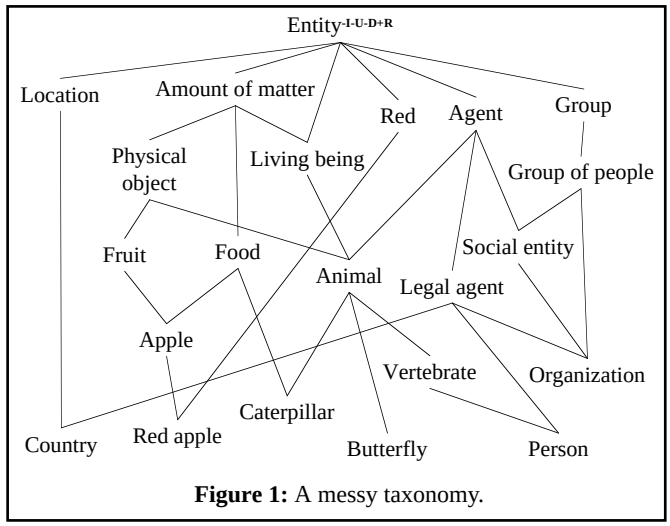

Figure 1: A messy taxonomy.

Assuming *ENTITY* has been already defined as shown, a dialog with the modeler might be:

```
What is the property name? amount-of-matter
What is the subsuming property? Entity
Is amount-of-matter rigid? (Y or N) Y
Does amount-of-matter supply identity? (Y,N,U) U
Does amount-of-matter carry identity? (Y,N,U) U
If an instance of amount-of-matter changes its parts, may it remain the same
instance? (Y,N,U) N
amount-of-matter carries identity.
amount-of-matter supplies identity.
Does amount-of-matter carry unity? (Y,N,U) U
Can instances of amount-of-matter be counted? (Y,N,U) N
amount-of-matter does not carry unity.
Is amount-of-matter dependent on any other properties? (Y,N) N
RESULT: amount-of-matter is +O+I+R-D-U
```

```
What is the property name? physical-object
What is the subsuming property? amount-of-matter
Is physical-object rigid? (Y or N) Y
Does physical-object supply identity? (Y,N,U) U
Does physical-object carry identity? (Y,N,U) U
If an instance of physical-object changes its parts, may it remain the same
instance? (Y,N,U) Y
VIOLATION: Non-mereologically extensional properties (physical-object) can not
be subsumed by mereologically extensional ones (amount-of-matter).
```

The modeler here has uncovered an inconsistency. A physical object, such as a car, can change some of its parts without becoming a different thing - we can change the tires of a car without making it a different car. An amount of matter, such as a lump of clay, is completely identified by its parts - if we remove some clay it is a different lump. A physical object is not, then, an amount of matter in this conceptualization, but it is *constituted* of matter. The relationship should therefore be one of constitution, not subsumption. The choice at this point is either to change the conceptualization of one of the two properties, or to change the taxonomic link.

The modeler chooses to put *PHYSICAL-OBJECT* below *ENTITY*, and continues, changing only the answer to the second question this time. We pick up the dialog from the last question:

```
If an instance of physical-object changes its parts, may it remain the same
instance? (Y,N,U) Y
Does physical-object have a characterizing feature that is unique to each
instance? (Y,N) U
Does physical-object carry unity? (Y,N,U) U
Can instances of physical-object be counted? (Y,N,U) Y
physical-object carries identity
physical-object supplies identity
physical-object carries unity
Is physical-object dependent on any other properties? (Y,N) N
RESULT: physical-object is +O+I+R-D+U
```

We now skip to the point where the modeler examines the class *ANIMAL*:

```
What is the property name? animal
What is the subsuming property? physical-object
Is animal rigid? (Y or N) Y
Does animal supply identity? (Y,N,U) U
Does animal carry identity? (Y,N,U) U
If an instance of animal changes its parts, may it remain the same instance?
(Y,N,U) Y
Does animal have a characterizing feature that is unique to each instance? (Y,N)
Y
What is the feature? brain
animal carries identity
animal supplies identity
Does animal carry unity? (Y,N,U) U
Can instances of animal be counted? (Y,N,U) Y
animal carries unity.
Is animal dependent on any other properties? (Y,N) N
```

```
Rigidity check: If an instance of animal ceases to be an instance of animal, does
it cease to be an instance of physical-object? (Y,N) N
VIOLATION: Rigidity check with physical-object failed.
```

The rigidity check question is only asked between rigid subsuming *sortals* (Def. 7). This question forced the modeler to think about the nature of the rigidity of this property. When an animal ceases to exist, i.e. when a person dies, their physical body remains. This indicates that, at least according to one conceptualization, *ANIMAL* is not subsumed by *PHYSICAL-OBJECT*, but as in the previous example, perhaps constituted of one.

To save space, we briefly describe the remaining changes instead of providing sample dialogs.

The analysis proceeds until all rigid properties in the taxonomy have been specified. *GROUP-OF-PEOPLE* carries the meta-property **+ME**, however *ORGANIZATION* does not, since people in organizations change, therefore this taxonomic link is removed. *SOCIAL-ENTITY* is also found not to have **+ME**, and therefore the taxonomic link to *GROUP-OF-PEOPLE* is removed from it as well. For similar reasons we remove the taxonomic link between *LIVING-BEING* and *AMOUNT-OF-MATTER*.

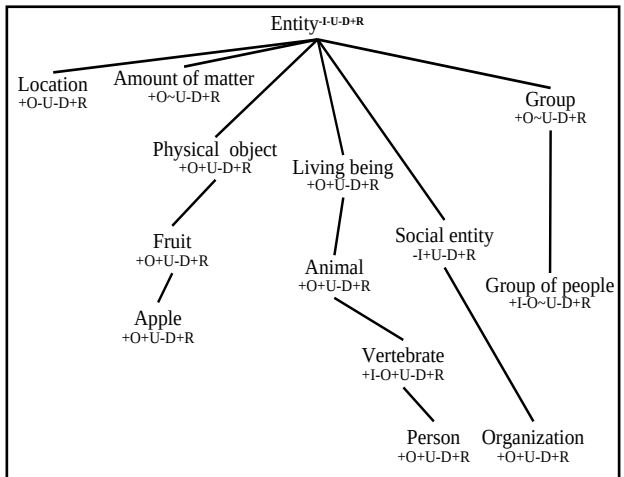

Figure 2: Primitive meta-property assignments for initially rigid properties.

The analysis of rigid properties continues through *LOCATION*, *GROUP*, *FRUIT*, *APPLE*, *VERTEBRATE*, and *PERSON*. The result of the meta-property analysis for the rigid properties is shown in Figure 2.

Once the rigid properties have been specified, we begin adding the non-rigid properties one at a time.

The property *AGENT* immediately causes problems because it is anti-rigid and subsumes two rigid properties (*ANIMAL* and *SOCIAL-ENTITY*). These taxonomic links are removed, which forces the modeler to consider why they were there. In this case, it is likely that a common misuse of subsumption as a sort of type restriction, such as “all agents must be animals or social entities,” was meant. It should be clear that logically, subsumption is not the same as disjunction, and another representation mechanism should be used to maintain this restriction.

The link between *FOOD* and *APPLE* must also disappear for the same reason (apple is not necessarily food), and *LEGAL-AGENT* is anti-rigid and can not subsume *PERSON* or *ORGANIZATION*. In this case, the link was being used as a type restriction, it was not the case that all *PERSON*s are *LEGAL-AGENT*s.

Analyzing *COUNTRY* brings us to an interesting case. *COUNTRY* may be, upon first analysis, anti-rigid, because a country, e.g. Prussia, may no longer exist, yet still be a place someone can go. Our deeper analysis reveals however, that two senses were being collapsed into one property: the sense of a geographical region, which is rigid, and a political or social entity, a country, which is also rigid (Prussia the country no longer exists, Prussia the region does). Again, countries are constituted of regions.

Analysis of *CATERPILLAR* and *BUTTERFLY* yields interesting examples. Closer inspection of these two properties reveals a special type of property, known as a *phased sortal* [29]. A phased sortal is a property whose instances can change from one sortal to another and still remain the same thing, i.e. a caterpillar becomes a butterfly, or in some systems, we can imagine that a student becomes an alumnus. Our methodology requires that phased sortals be identified along with all the corresponding phases, and grouped under a rigid property that subsumes *only them* [15]. We add, therefore, *LEPIDOPTERAN*.

Finally, the property *RED* is non-rigid because, although most things are not necessarily red, there may be some things that are. Non-rigid properties make very little commitment, and are often confusing parts of a taxonomy. In this case, it is used to distinguish *RED-APPLE* from presumably other color apples.

With the analysis of all properties complete, our technique identifies the backbone taxonomy and the final cleaned taxonomy, shown in Figure 3. Note that one result of this "cleaning" process is the removal of many occurrences of multiple inheritance. This is not necessarily a specific goal, however it naturally follows from the fact that, as discussed in [16], multiple inheritance is often used as a tool to represent more than simply subsumption – as we found in this example. We believe that these cases make taxonomies confusing; if the purpose of an ontology is to make the meaning clear, then the meaning should not be clouded by using the same mechanism to signify more than one thing, since there is no way to disambiguate the usage. Furthermore, there is at least some empirical evidence derived from studies of programmers who maintain object-oriented programs that multiple inheritance is confusing and makes taxonomies difficult to understand [18].

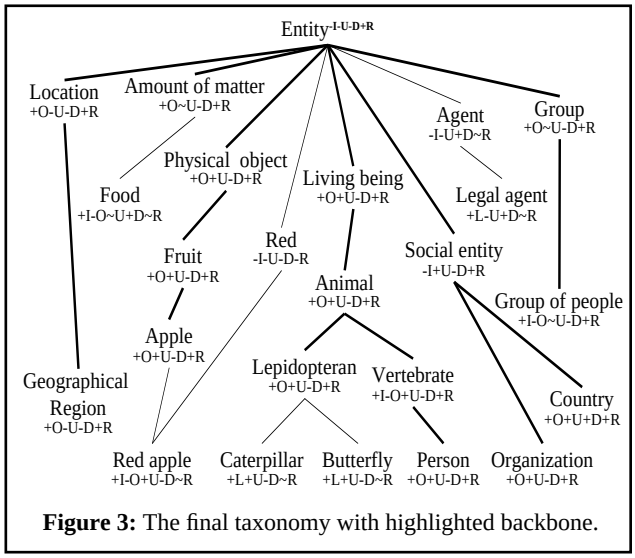

Figure 3: The final taxonomy with highlighted backbone.

6 Conclusion

We have discussed several notions of Formal Ontology used for ontological analysis in Philosophy: identity, unity, essence, and dependence. We have formalized these no-

tions in a way that makes them useful for conceptual modeling, and introduced a methodology for ontological analysis founded on these formalizations.

Our methodology is supported by a question/answer system that helps the conceptual modeler study the deep ontological issues surrounding the representation of properties in a conceptual model, and we have shown how this methodology can be used to analyze individual taxonomic links and make the taxonomy more understandable. In particular, we have also shown how to identify the backbone taxonomy, which represents the most important properties in an ontology that subsume every instance.

Unlike previous efforts to clarify taxonomies, our methodology differs in that:

- It focuses on the nature of the properties involved in subsumption relationships, not on the nature of the subsumption relation itself (which we take for granted).
- It is founded on formal notions drawn from Ontology (a discipline centuries older than database design), and augmented with practical conceptual design experience, as opposed to being founded solely on the former or latter.
- It focuses on the validation of single subsumption relationships based on the *intended meaning* of their arguments in terms of the meta-properties defined here, as opposed to focusing on structural similarities between property descriptions.

Finally, it is important to note again that in the examples we have given, we are providing a way to make the *meaning* of properties in a certain conceptualization clear. We do not, for example, mean to claim that "Person is-a Legal-Agent" is wrong. We are trying to point out that *in a particular conceptualization* where *LEGAL-AGENT* has certain meta-properties (such as being anti-rigid) and *PERSON* certain others (such as being rigid), it is inconsistent to have person subsumed by legal-agent.

References

1. Artale, A., Franconi, E., Guarino, N., and Pazzi, L. 1996. Part-Whole Relations in Object-Centered Systems: an Overview. *Data and Knowledge Engineering*, **20**(3): 347-383.
2. Bergamaschi, S. and Sartori, C. 1992. On Taxonomic Reasoning in Conceptual Design. *ACM Transactions on Database Systems*, **17**(3): 285-422.
3. Brachman, R. 1983. What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks. *IEEE Computer*, **16**(10): 30-36.
4. Brachman, R. J., McGuinness, D. L., Patel-Schneider, P. F., Resnick, L., and Borgida, A. 1990. Living with CLASSIC: When and How to Use a KL-ONE-like Language. In J. Sowa (ed.) *Principles of Semantic Networks.* Morgan Kaufmann: 401-456.
5. Calvanese, D., Lenzerini, M., and Nardi, D. 1998. Description Logics for Conceptual Data Modeling. In J. Chomicki and G. Saake (eds.), *Logics for Databases and Information Systems*. Kluwer: 229-264.
6. Elmasri, R., Weeldreyer, J., and Hevner, A. 1985. The category concept: An extension to the Entity-Relationship model. *Data and Knowledge Engineering*, **1**(1): 75-116.
7. Goldstein, R. C. and Storey, V. C. 1999. Data abstractions: Why and how? *Data and Knowledge Engineering*, **29**: 293-311.
8. Gruber, T. R. 1993. Model Formulation as a Problem-Solving Task: Computer-Assisted Engineering Modeling. *International Journal of Intelligent Systems*, **8**: 105-127.
9. Guarino, N. 1992. Concepts, Attributes and Arbitrary Relations: Some Linguistic and Ontological Criteria for Structuring Knowledge Bases. *Data & Knowledge Engineering*, **8**(2): 249-261.

10. Guarino, N., Carrara, M., and Giaretta, P. 1994. An Ontology of Meta-Level Categories. In D. J., E. Sandewall and P. Torasso (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94)*. Morgan Kaufmann, San Mateo, CA: 270-280.
11. Guarino, N. 1995. Formal Ontology, Conceptual Analysis and Knowledge Representation. *International Journal of Human and Computer Studies*, **43**(5/6): 625-640.
12. Guarino, N. 1998a. Formal Ontology in Information Systems. In N. Guarino (ed.) *Formal Ontology in Information Systems*. Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998. IOS Press, Amsterdam: 3-15.
13. Guarino, N. 1999. The Role of Identity Conditions in Ontology Design. In *Proceedings of IJCAI-99 workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends*. Stockholm, Sweden, IJCAI, Inc.
14. Guarino, N. and Welty, C. 2000a. Identity, Unity, and Individuality: Towards a Formal Toolkit for Ontological Analysis. In *Proceedings of ECAI-2000: The European Conference on Artificial Intelligence*. Berlin, Germany, IOS Press. Available from http://www.ladseb.pd.cnr.it/infor/ontology/Papers/OntologyPapers.html
15. Guarino, N. and Welty, C. 2000b. A Formal Ontology of Properties. In Rose Dieng (ed.), *Proceedings of 12th Int. Conf. on Knowledge Engineering and Knowledge Management*, Springer Verlag. Available from http://www.ladseb.pd.cnr.it/infor/ontology/Papers/OntologyPapers.html
16. Guarino, N. and Welty, C. 2000c. Towards a methodology for ontology-based model engineering. In *Proceedings of ECOOP-2000 Workshop on Model Engineering*. Cannes, France. Available from http://www.ladseb.pd.cnr.it/infor/ontology/Papers/OntologyPapers.html
17. Hirst, G. 1991. Existence Assumptions in Knowledge Representation. *Artificial Intelligence*, **49**: 199-242.
18. Huitt, R. and Wilde, N. 1992. Maintenance Support for Object-Oriented Programs. *IEEE Trans. on Software Engineering*, **18**(12).
19. Humberstone, I. L. 1996. Intrinsic/Extrinsic. *Synthese*, **108**: 205-267.
20. Lowe, E. J. 1989. *Kinds of Being. A Study of Individuation, Identity and the Logic of Sortal Terms*. Basil Blackwell, Oxford.
21. Quine, W. V. O. 1969. *Ontological Relativity and Other Essays*. Columbia University Press, New York, London.
22. Simons, P. 1987. *Parts: a Study in Ontology*. Clarendon Press, Oxford.
23. Storey, V. C. 1993. Understanding Semantic Relationships. *Very Large Databases Journal*, **2**: 455-488.
24. Storey, V., Dey, D., Ullrich, H., and Sundaresan, S. 1998. An ontology-based expert system for database design. *Data and Knowledge Engineering*, **28**: 31-46.
25. Strawson, P. F. 1959. *Individuals. An Essay in Descriptive Metaphysics*. Routledge, London and New York.
26. Teorey, T. J., Yang, D., and Fry, J. P. 1986. A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model. *ACM Computing Surveys*, **18**(2): 197-222.
27. Welty, C. *A description-logic based system for ontology-driven conceptual analysis.* System demo available at http://untangle.cs.vassar.edu/odca/.
28. Wieringa, R., De Jonge, W., and Spruit, P. 1994. Roles and dynamic subclasses: a modal logic approach. In *Proceedings of European Conference on Object-Oriented Programming*. Bologna.
29. Wiggins, D. 1980. *Sameness and Substance*. Blackwell, Oxford.