An ontological approach for reliable data integration in the industrial domain

Article *in* Computers in Industry · February 2014

DOI: 10.1016/j.compind.2013.12.010

1 author:

Stefano Borgo
Italian National Research Council
162 PUBLICATIONS **3,837** CITATIONS

S. Borgo "An ontological approach for reliable data integration in the industrial domain." Computers in Industry, 65(9):1242–1252, 2014.

An ontological approach for reliable data integration in the industrial domain

Stefano Borgo

Laboratory for Applied Ontology, ISTC CNR
via alla cascata 56C, 38123 Trento, Italy
stefano.borgo@cnr.it

Abstract

Ontologies are structural components of modern information systems. The taxonomy, the core of an ontology, is a delicate balance between adequacy considerations, minimal commitments and implementation concerns. However, ontological taxonomies can be quite restrictive and entities that are commonly used in production and services might not find room in a official or de facto standard or ontological system. This mismatch between the company's view and the ontological constraints can limit or even jeoparize the adoption of modern formal ontologies in industry. We study the roots of this problem and individuate a general set of principles to relate the ontology and those non-ontological entities that are yet important for the core business of the company. We then introduce a theoretically sound and formally robust approach to expand a given ontology with new dependency relations, which make available information regarding the non-ontological entities without affecting the consistency of the overall information system.

Keywords: Formal ontology, Product data, Design data, Information integration

1. Introduction

Imagine you pick a book at a bookstore, read parts of it, find it interesting and finally buy the book. Any learned ontologist and knowledge engineer would distinguish at least two types of entities here: a physical entity (the physical-book that is bought at the bookstore and carried home in a bag) and an information entity (the information-book that that changes the knowledge or state of mind of the reader). The physical vs. information (or functional) distinction, blurred in natural language, is only one of the targets of ontological analysis and is frequently undetected in everyday life. Yet, it helps to answer questions like: what do we pay for when we by a stone sold as a paperweight? why do we recognize cars in a wreckage deposit?

Modeling domain knowledge in an ontologically consistent way is a fascinating enterprise and pushes the knowledge engineer to deal with a series of important distinctions that go often undetected in commonsense as well as in professional life. The goal is to model information as used in a domain application while avoiding possible

sources of confusion and ambiguities. The problem we address in this paper is part of the large effort to improve today's organization and management of the information generated in engineering design and manufacturing and, more generally, in the production life cycle at large. More specifically, let us call *domain entities* the things a company talks about in its everyday business. It is well known that some of these entities may be ontologically unsound, the goal of this paper is to find a methodology that allows to recognize and process information regarding these domain entities even if not consistent with the adopted ontology. If we succeed, give the possibility to companies to embrace modern information systems, and their underlying ontologies, without giving up its own language and, more importantly, business perspective.

We started with the classical example of the book as a physical thing vs the book as an information object. This example is quite intuitive and has been analyzed at length in the literature. In linguistic semantics notions like *constructional polysemy* and *co-predication* [2] have been proposed to make sense of the double meaning of the term book. Semanticists analyze the use of these terms as an indication of the need for combinations of categories in different branches linguistic ontology. From this view, an expression like "That book is 500 pages long and is difficult to read" [2, p.257] is a case of category overdetermination and special constructs are proposed to justify the existence of mixed categories whose elements collect all the needed properties. The idea is that an expression like "This car runs 120 mhp and is selling well" should be understood as talking about a new type of object which *combines* a physical object (with its physical properties), and a type (to which the statistics on the selling events on the market are referred). The distinction between physical and abstract or between instance and type is essential but the proposal to introduce mixed categories is not acceptable in ontology since incompatible properties like "being abstract" (property of the book content) and "being concrete" (property of the physical book) cannot coexist in the same entity. This shows that the linguistic approach is not a solution for our case.

Since the mixture of incompatible properties often result in logical inconsistency, one can see if logicians have found a suitable way to deal with the physical vs. information (and analogous) distinctions. In formal logic, one aks whether the term 'book' is actually denoting something. Logic allows to choose between three options: the term does not denote, it denotes one entity or it denotes more than one entity [36, Chp. 8]. We have already seen the problems raised by assuming it denotes just one thing. Instead, in *free logic* a term may have no denotation at all. Roughly speaking, this view says that the notion of book does not correspond to a real entity. Although formally correct, this is not an ontologically suitable solution since it is hard for a book producer to believe that what she calls book, her business object for which she makes models and production plans, is a non-existing entity. The third option, a term can have multiple denotations, is the view taken in paraconsistent logic [36, p. 160]. The knowledge engineer might be satisfied with multiple denotations since all domain entities, including books and cars, are recognized as meaningful under this perspective. Furthermore, even the ontologist would agree with this view since it allows to distinguish domain entities that are ontologically unsound: indeed *ontologically sound entities have a unique denotation while the others do not*. Unfortunately, the multiple denotation approach makes the relationship between domain entities cumbersome: in paraconsistent logic

identity is not transitive and even basic logical principles, like *particular generalization*, fail. An information system based on this view might conclude that the same product A, which has label $HD\text{-}Id32$ for the handling department and $PD\text{-}Id121$ for the production department, may very well be two different things leading to unacceptable errors in counting how many products are in stock.

We have seen that natural language and logic propose unsatisfactory solutions to our problem and, to the best of our knowledge, no better solution comes from other domains. We believe this is due to the fact that the problem we are after is a truly ontological problem and we are now going to discuss it in these terms.

The paper is structured as follows. Section 2 introduces ontological taxonomies and explains why they are essentially tree-shaped. Section 3 clarifies the role of the ontology module in an information system and concludes that some entities are necessarily left out. Section 4 discusses the principles on which our approach relies. Section 5 briefly introduces two ontologies exploited in the industry domain and used later to provide examples. Section 6 presents our methodology divided in four phases and gives examples for each. Finally, Section 7 adds further observations and points to future work.

2. Taxonomies and Criteria

An information system based on an ontology adopts, at the minimum, the taxonomic structure of the ontology, that is, the basic hierarchy of categories (aka concepts, types, classes)[1] and the criteria for category membership. The taxonomy impacts issues like the (types of) entities that can possibly exist, the properties they share, how entities and properties are related, and which relationships hold among entities and among properties.

Ontologists have exploited taxonomies in different formats. There are essentially two ontological structuring relations suitable to building articulated taxonomies, namely the relation of *part* and the relation of *subsumption* (or *ISA*). Other ontological relations like *participation* and *causation* do not partition the domain, or are non-iterative like the *instance-of* relation. Furthermore, relations that are linguistically (or even psychologically) motivated like *similar-to*, *opposite-of* and *kind-of*, have ambiguous interpretations and thus are ontologically unclear. Since parthood is actually a collection of different relations [41, 46, 27] that are not easy to keep apart, practically any ontological system of broad scope relies primarily on the subsumption relation. We will thus focus on these subsumption-based systems.

In principle, ontological taxonomies can use subsumption in different ways [45]. The taxonomies used in industrial applications are mostly *tree*-shaped, we will see an example of this in Section 5.2. In most of the other cases the requirement about disjointedness of subcategories is relaxed, which allows a category to be a specialization

[1] These terms are often associated with different meanings, which however vary across research communities. In ontology concepts are mostly understood as mental entities, types as abstractions from individuals, classes as extensions of properties, and categories as ontologically motivated classifiers. In this paper these distinctions are not crucial, thus we take these terms as synonyms.

of two or more larger categories. This phenomenon is known as *multiple inheritance*. Technically, these taxonomies have the structure of *directed acyclic graphs* (DAG), we will see a case in Section 5.1.

The preference for tree-shaped taxonomies is grounded in a series of methodologies, e.g., OntoClean [25] and variants like [40, 47], which analyze the ontological meta-properties of categories and guide a coherent subsumption implementation. For instance, OntoClean motivates the substitution of the top taxonomy in Fig. 1 with the taxonomy on the bottom.

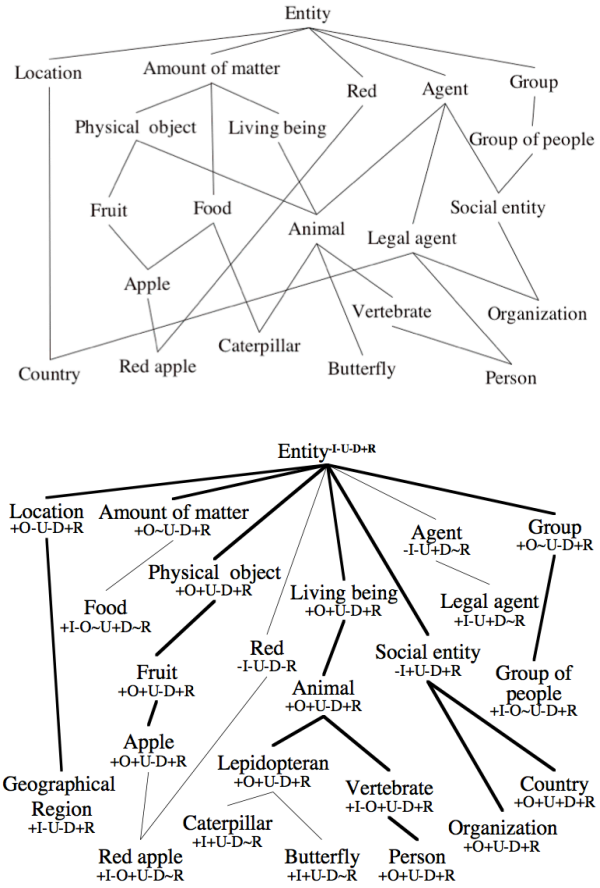

Figure 1: A taxonomy before (top) and after (bottom) the application of OntoClean [25, p.219]. (The new labels on the right taxonomy point to category properties that motivate the changes.)

Ontological taxonomies implement a series of principles among which the following:

> First, a category system should strive to be *exhaustive*, providing a *complete* list of highest kinds so that there is a category for everything there

> might be. [...] The categories provided (at any given level) should be mutually exclusive, so that we avoid redundancy (and retain efficiency) and ensure that whatever there is can be uniquely located in exactly one category. [45, p.9]

The first requirement for *structural adequacy*, known as the principle of *classification completeness* ("there is a category for everything there might be"), is particularly hard to achieve and has an important consequence: any entity not belonging to one of the categories in the chosen ontology, does not exist (relatively to that ontology). The second requirement, the principle of *ontology clearness*, amounts to say that ontological categories are disjoint.

Ontological taxonomies are also bound to more philosophical principles called ontological commitments. Two criteria are due to Quine [37, 38]: *ontological minimality* and *ontological adequacy*. According to the first criterion the ontology should commit to the existence of a smallest number of categories. The selection of the categories to include in the ontology follows some *respectability* guidelines, where respectability is the result of ontological considerations; for instance, the possibility of entities that do not have clear identity criteria, is ontologically questionable and thus these kinds of entities should not be accepted. The minimality criterion has been considered by many authors although with different motivations as in the following snippet:

> An ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. An ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology as needed. [22, p. 910]

The *ontological adequacy* criterion says that the domain of interest should be modeled in its integrity: in particular the principle leads to include all entities that are referred to in the domain of interest.

We have seen that taxonomical structures in ontology are based on general principles which, although well motivated, can be hard to implement and, in complex domains, may lead to conflicting decisions as, e.g., the criteria of minimality and adequacy. This difficulty is in our view the theoretical source of several problems discovered in ontology-based information systems and exemplified in studies like [1], see also the gaps 8 and 9 in [23].

3. Ontologies and Domain Entities: the Problem

Assume we want to extend a given ontology with a category that corresponds to a product tupe as modeled in a company database; say we want to deal with a type of printed circuit board populated with electronic components, i.e. a Printed Circuit Board Assembly (PCBA), like those in modern electronic devices: from computers to remote controls, from cameras to turbine control units.

According to the guidelines [24], one first verifies the ontological status of the category (corresponding to the database class) and lists its identity criteria: what characterizes this category? how to identify entities that belong to it? which properties are shared by its members? Next, one tries to add the new category to the taxonomy as a specialization of an existing category [32, 11]. A variety of techniques for ontology construction come helpful especially when dealing with large ontologies [44, 33, 48] or when aligning ontologies (perhaps with a database) [15, 34, 39, 35, 49, 17]. Unfortunately, known techniques assume that the new category is *ontologically* (or even conceptually) compatible with the given ontology or that it is acceptable to change the meaning of the category. Our class of PCBAs does not fall within these cases as we are going to see.

Since we are looking for a general approach, it is important to keep in mind the role of an ontology in an information system and to refrain from *ad hoc* solutions. Generally speaking, an ontology *stipulates* what can possibly exist from the system's perspective. That is, it establishes, with no room for exceptions, which entities the system *can* identify and classify. Furthermore, the ontology provides the (articulated) catalog of all *types of entities* whose actual or just possible existence is admitted by the system. For instance, if an ontology allows only things that are in time *and* in space (spatio-temporal entities), then any information system based on it cannot include, say, `deadline for delivery` as an abstract entity but has to model it as something that has spatial and temporal qualifications, e.g., as a property of a spatio-temporal event `delivery` with properties `time of delivery` and `location of delivery`.

Is the category of PCBA incompatible with an ontology? The PCBA class in a company database collects a series of data relative to an identifier, e.g., time of production, base material, size of the board, structure of the board (e.g. single, double sided, multilayers), type(s) of interfaces, number and types of electronic components, functionalities, result of the quality test(s), price, and so on. Some of these fields can be simple like unstructured alphanumeric strings, others may unfold into rich sub-structures, e.g., a PCBA may have several functionalities explicitly structured in hierarchies or implicitly encoded via reference to compatibility with other PCBA types. Some properties of the PCBA belong to the physical structure of the described product like the time of production, the size of the board, the number of components and data from the bill of materials. Others, like functionalities, result of quality test, compatibility and price, describe how the product behaves or is valued in relation to other entities. Clearly the physical-product and the behavioral (or functional)-product are distinct entities from the ontological viewpoint. One has different options to model behavior and function (is it a pattern? a relation? an abstract entity? an information object?) but for sure it is neither a physical object nor a simple property.[2] Since the distinction between *internal properties*, e.g. physical properties, and *external properties*, e.g. functionalities, is crucial in ontologies in general, from foundational ontologies like DOLCE [30], standards based on ontology like ISO 15926-2 [26], and general purpose ontologies

[2]The foundamental distinction between the physical and functional aspects of products is also recognizable from other viewpoints, see e.g. the structural and abstraction hierarchies in [21].

like CYC[3], we conclude that the PCBA category is ontologically incompatible with most ontologies in use today.

Thus, as of today, a company that wants to adopt an ontology or a modern standard might be required to give up some of its core notions like the PCBA category or, for a different example, the notion of schematic (technical) drawing. A schematic drawing is an entity that has properties associated to disjoint ontological categories: on the one hand properties typical of physical objects like drawing (the creation act) date/time, drawing size, drawing coloring, number of marks etc.; on the other hand properties typical of information objects, a subcategory of concepts in [8], like the meaning of marks, scale, representation of projective views, and so on.

Before explaining how we are tackling the problem, we introduce a few definitions.

Let us call *ontology* any logical theory based on a vocabulary $\leqslant, C_1, \ldots, C_n, R_1^{s_1}, \ldots, R_m^{s_m}$ where $\leqslant$ (informally, the subsumption relation) is a partial order on $C_1, \ldots, C_n$; the C_i's are unary predicates (the predicates for the categories); and each $R_j^{s_j}$ is a relation of arity s_j (the properties and relations in the ontology). We require that $n \geqslant 1$ and $m \geqslant 0$, that is, there must be at least one category in the ontology. To minimally constrain the use of the vocabulary, we add the following (in the language of first-order logic):

(i) if $C_h \leqslant C_k$ then $\forall x(C_h(x) \rightarrow C_k(x))$;

(ii) $\forall x(\bigvee C_i(x))$;

Condition (i) says that $\leqslant$ is the subsumption relation. The second condition formalizes the first criteria of structural adequacy and says that the categories $C_1, \ldots, C_n$ cover the whole domain. These are the constraints on which we will rely.

The second criteria of structural adequacy, the principle of ontology clearness, is formally stated as follows but is not enforced:

(iii) if $C_i \cap C_j \neq \emptyset$, then either $C_i \leqslant C_j$ or $C_j \leqslant C_i$.

The condition states that two categories are either disjoint or one is a specialization of the other. Other constraints, like the existence of a root category, are not necessary for our argument.

Let $\mathcal{O}$ be an ontology, i.e. a logical theory based on a vocabulary as just described and satisfying conditions (i) and (ii). We call *ontological entities* the entities that are classified by $\mathcal{O}$, i.e. those whose existence was intended by the ontology designer. Recall from the introduction that we call *domain entities* the entities that are motivated by the application domain. Clearly, depending to the chosen ontology $\mathcal{O}$, many domain entities may also be ontological entities, that is, are recognized and classified by $\mathcal{O}$. Other domain entities, as PCBA's and schematic drawings, may not be accepted by $\mathcal{O}$. Let us call *extra-ontological entity* relatively to an ontology $\mathcal{O}$ any domain entity that is not classified by $\mathcal{O}$, i.e. that does not belong to any category of $\mathcal{O}$.

[3] www.opencyc,org

From the viewpoint of the ontology engineer, extra-ontological entities can be divided in two types. On the one hand, the extra-ontological entities that have properties belonging to two (or more) disjoint ontological categories, like our PCBA. These entities are conceptually, and thus logically, incompatible with the ontology. On the other hand, the extra-ontological entities that *prima facie* belong to some ontological category of $\mathcal{O}$ but are not properly described by it. Typically, this happens because some crucial property of the entity is not captured in the language of $\mathcal{O}$. An example is given by a device A that is explicitly designed for usage in highly inflammable environments like an oxygen chamber, but the ontology that should classify A cannot talk about safety requirements. $\mathcal{O}$ correctly classifies the entity A as a device but misses some essential property of A. Entities like A are compatible with the ontology both at the conceptual and at the logical level; they fall within the ontology alignment area we mentioned earlier (see also [5]) and are out of the scope of this paper.

From now on we assume that any property and relation needed to model a domain entity is readily available in the language of the chosen ontology $\mathcal{O}$, i.e., that the C_i and R_j's in the language of $\mathcal{O}$ can fully characterize both the ontological and the extra-ontological entities at stake.

4. The Theorethical Guidelines

This section exploits the work on dependence relations developed in [16] and [13] by applying general principles that examine how ontological properties and extra-ontological entities can be related. If there is an ontological mismatch between the entities in the domain, say, as cast in a company's database, and those in the ontology, we want a rigorous and logically consistent way to relate them.

Generally speaking, we have two options: (a) to integrate the database and the ontology into a suitably adapted knowledge base; (b) to integrate the database schema and the ontology into an ontologically sound system. Since we are looking for a general method and are not concerned with *ad-hoc* solutions, we take the second option. Note that we ignore here other aspects like those related to the formal language (aiming to remain neutral, our examples will be given in first-order logic) and reasoning systems.

We say that a category D is *extra-ontological for ontology* $\mathcal{O}$ whenever

(a) the language of $\mathcal{O}$ can express all the essential properties $R_1, \ldots, R_k$ that a member of D must satisfy,

(b) for each R_i ($1 \leqslant i \leqslant k$) there is at least one category C_{j_i} of $\mathcal{O}$ whose members must satisfy R_i and[4]

[4]For the ontologically inclined reader let us observe that this requirement is necessary. For a general example, consider the abstract category of triangles (in a metric space) and an ontology admitting only physical objects and events, i.e. not including abstract entities. Without the constraint, one could use the methodology we describe in Section 6 to relate the abstract category of triangles to the ontology by ignoring the abstractness property. The obtained relationship would be correct from the logical viewpoint but not from the ontological one since this ontology excludes the possibility to instantiate abstractness.

(c) $\mathcal{O}$ assumes that there cannot exist entity x such that $\bigcup_{1 \leqslant i \leqslant k} R_i(x)$.

Given an ontology $\mathcal{O}$, ontological categories $C_1, \ldots, C_n$ of $\mathcal{O}$ and an extra-ontological category D for $\mathcal{O}$, we call *patch relationship* a relation $Patch$ such that for some $k \geqslant 2$

$$\forall x \exists y_1, \ldots, y_k (x \in D \rightarrow \bigcup_i C_{j_i}(y_i) \wedge Patch(x, y_1, \ldots, y_k)). \tag{1}$$

According to formula (1), $Patch$ is a relation between a domain category D and two or more ontological categories C_{j_i} such that for any element x of D, there exist elements y_i of C_{j_i} for which $Patch$ holds with arguments $x, y_1, \ldots, y_k$. When such a relation holds, we say that D *is patched to* $\mathcal{O}$, and that D and C_{j_i} (for each i) are *patched*.
Note that the patch relation does not lead to include category D into the ontology $\mathcal{O}$, after all these formulas are not even formulas in the ontology language: they quantify over the mixed domain of $\mathcal{O}$ and of D. The role of the patch relations is made clear by the following principles on which our approach relies:

Principle 1 (Patchable Categories). *Given an ontology $\mathcal{O}$, there can exist categories which are extra-ontological for $\mathcal{O}$. The ontology and each such extra-ontological category can be associated via a patch relationship.*

This principle claims that the system formed by $\mathcal{O}$, the extra-ontological categories D's and the patch relationships are ontologically meaningful. In other terms, it leads to admit that something can exist outside the ontology.

Principle 2 (Ontology Perspective). *From the perspective of $\mathcal{O}$, an extra-ontological category D is a dependency relationship among ontological categories (or better, ontological entities) in $\mathcal{O}$.*

That is, the ontology $\mathcal{O}$ does not recognize the extra-ontological categories as ontological categories. It however recognizes the dependences among ontological categories (or ontological entities) introduced *within* the ontology $\mathcal{O}$ by the patch relationships.

Principle 3 (Patch Dependence). *Let D be an extra-ontological category for $\mathcal{O}$ and $C_1, \ldots, C_n$ the categories of $\mathcal{O}$ to which it is patched. Then, for any entity x in D there exist an entity y_i in C_i such that the patch holds for tupla $< x, y_1, \ldots, y_n >$.*

This principle states that ontological and extra-ontological categories, when patched, are bound at least to the minimal form of existential dependence given by formula (1).

Principle 4 (Patch Directionality). *Let D be an extra-ontological category for $\mathcal{O}$ and let C be one of the categories of $\mathcal{O}$ patched to D. There can be an entity in C which does not occur in any instance of the patch relationship.*

Let D and $C_1, \ldots, C_n$ be patched and let $Patch_D(x, y_1, \ldots, y_n)$ be the patch relationship among them. Principle 4 states that for some i there might exists $y' \in C_i$ such that for all $z \in D$ and all $y_j \in C_j$ ($j \neq i$), then $R(z, y_1, \ldots, y', \ldots, y_n)$ fails. In other terms, while Principles 2 and 3 say that the introduction of the extra-ontological

category D has consequences on the ontology at the level of dependences among ontological entities, Principle 4 says that the opposite may not hold: the ontology has not impact on the extra-ontological category by the introduction of the patch relationship.

To justify the introduction of this latter principle, consider the category of physical representatives of lengths. An instance of this category is the physical meter which has been *the* object officially measuring 1 meter in length until 1983. If we are using an ontology that adopts the laws of physics as we know them today, there are more lengths than physical objects. Actually, there are lengths for which no physical object can exist, due to limits in the size of the universe. Then, Principle 4 prevents the ontology from assuming the actual existence of a physical object for each theoretically possible length.

In this section we have listed the principles that will guide the application of dependency theory to our information integration problem. These statements are called principles since they: (a) fill the gap between purely abstract work and implementation in information theory; (b) are independent from the categories of the ontology and of the application domain; (c) apply to any formal language and information system that can express relational properties. (Dependences may be limited in the case of restricted languages like, e.g., RDF(S) or description logics [3].)

The next step is to provide a methodology based on these principles but first we introduce two ontologies used in industrial applications. These will be exploited to exemplify applications of the methodology.

5. Two Ontologies for the Engineering Domain

The next section provides a methodology, based on the principles of Section 4, that we propose to solve the problem described in Section 3. The method consists of four phases which we describe in detail and exemplify by referring to the categories of PCBA and that of PCBA schematic drawing (SD). To show the generality of the method, we apply it to two quite different ontologies: the conceptual data model ISO 15926-2 [26, 4], which was initially motivated by modeling concerns in construction and operation of oil and gas production facilities but is today applied to other domains as well [28, 14]; and the foundational ontology DOLCE [30], which is used in areas like engineering design methodology [6, 20], inventive design [50] and manufacturing [10, 7]. For reviews on ontologies and semantic standards see [1, 43, 7, 18] and references therein.

5.1. ISO 15926-2

The data model of the standard ISO 15926-2 [26] has a top-level that is claimed to be a full-fledged ontology [4]. The standard is the result of an effort started in 1992 within the EU ESPRIT project "ProcessBase". The goal of ISO 15926-2 is data integration, sharing, exchange, and hand-over between information systems and it is fair to say that the standard goes well beyond its initial title, i.e., "Industrial automation systems and integration–Integration of life-cycle data for process plants including oil and gas production facilities". The data model of ISO 15926-2 (Part 2), based on EPISTLE, was completed 2003.

ISO 15926-2 top-level divides entities in two categories: ABSTRACT_OBJECT and POSSIBLE_INDIVIDUAL. The first collects entities like numbers and sets, the latter entities that have a spatio-temporal extension. This second category embraces a vision called *four dimensionalism* where an expression like "I kicked the door" has to be understood as, roughly, "at some point in time the spatio-temporal process known as my foot violently interacted with the spatio-temporal process known as the door". In other terms, members of this class are not objects in the naïve sense but processes; for another example, 'possessing an object' means that a person's spatio-temporal process has the (social) right to affect the object's spatio-temporal process evolution. Thus, from the ISO 15926-2 perspective, reality is a combination of processes and, leaving aside abstract entities, existence is restricted to processes in space-time.

ISO 15926-2 divides the category of processes called POSSIBLE_INDIVIDUAL into subcategories like ARRANGED_INDIVIDUAL, e.g. a car (a process) is an arranged individual since it has components (also processes) with different roles like the engine and the shaft; ACTUAL_INDIVIDUAL, e.g. my desk (a process) is an actual individual since it happens in the actual world and not in an imaginary or possible world; WHOLE_LIFE_INDIVIDUAL, e.g. a screwdriver (a process) is in this category since it is not part of another screwdriver (contrast this to a land which is itself part of a land, or the Adriatic sea which is part of the Mediterranean sea); PHYSICAL_OBJECT, e.g. a pump (again, a process); etc. These categories are exhaustive, according to the first structural principle discussed in Section 2, but are not exclusive: the (spatio-temporal process of the) pump can be at the same time an actual individual, an arranged individual and a whole life individual.

Inspired by the modeling of information relative to the whole life-cycle of technical facilities and their parts, ISO 15926-2 can be applied to generic domains to model information about functional requirements, physical realizations, objects and activities classifications, etc. ISO 15926-2 is available in first order logic and in OWL (web ontology language) [4].

Problems with ISO 15926-2: the ISO 15926-2 ontology gives a concrete example of the limitations provided by the exhaustiveness principle (Section 2). As said, in this view a car is a process: it starts with the car construction, evolves with the different uses of the car and ends with the destruction of the car. In particular, only a temporal part of the car is present at any point in time since by 'car' ISO 15926-2 means the entire process, i.e., a whole space-time worm. Although there is nothing wrong with this view, from our experience engineers in the production domain and providers of PCBA refurnishing services see a PCBA as a physical object in the standard (three-dimensional) naïve sense, i.e., an entity that is completely present when the factory hands it. The claim "we produce 30 PCBA of type $\#234$ per day" means that every day there are 30 new objects of a given type ready to ship. The corresponding view that there are 30 initial parts of 30 new processes that have some given characteristics and that are ready to continue their evolution detached from the factory evolution is not wrong per se but definitely not a natural perspective.

5.2. DOLCE

The *Descriptive Ontology for Linguistic and Cognitive Engineering* (DOLCE) [31, 8] is a foundational ontology introduced in 2002 and applied in domains like enterprise modeling, product modeling, financial transactions, medicine, geographic information systems, and the semantic web. For an evaluation of the ontology see [42]. DOLCE is a formal system aiming to model categories that are general and widely applicable. The ontological setup of the system and the categorical structure are based on philosophical principles the choice of which is explicitly motivated [31]. DOLCE takes a common-sense perspective and is particularly suitable for domains where one relies on classical physics and the human-centric viewpoint.

The DOLCE ontology distinguishes particulars from universals. Roughly speaking, a *universal* is an entity that is instantiated by other entities (like "being human" and "being a car"). A *particular* is an entity that is not instantiated by other entities like the Eiffel Tower in Paris and the laptop with which this paper has been written. Particulars comprise also entities that happen in time like events (the delivery of item ID123) and processes (the running of machine MC321); in particular it adopts the usual distinction between objects like products, and events like operations. Individual qualities (properties of a given particular) are also particulars.

Following the terminology in [8], the category OBJECT collects entities like a *hammer*, and amounts of matter, like *the amount of plastic that is in my cell phone*, while the category EVENT has as elements entities that happen in time like *making a hole* or *designing a screwdriver*. Both objects and events are associated with a bunch of individual qualities (elements of the category QUALITY). The exact list of qualities depend on the entity: *shape* and *weight* are usually taken as qualities of objects (physical qualities), while *duration* is a quality of events (temporal quality). An individual quality is a quality associated with *one and only one* particular; it can be understood as the special way in which that entity instantiates a general property. For example, the weight of the chair I am sitting on now is an individual quality of that unique chair and instantiates the property "having weight" relative to that chair. The change of an object in time is explained through the change of some of its individual qualities. For example, the object Drill#321 (say, the one I own) has its own individual instantiation of property "having weight", i.e., the individual weight-quality of Drill#321. When a component is substituted, damaged or anyway altered, Drill#321 may change its weight so that the individual weight-quality of Drill#321 can be assigned to different values as time passes.

Problems with DOLCE: the DOLCE ontology allows to exemplify the limitations of the second ontological principle of Section 2, namely, category exclusivity. DOLCE clearly divides physical objects and information objects but in the industrial domain we deal with schematic drawings of devices which, in the standard view of people working in the domain, are both physical entities (perhaps stored in a database) and information entities: see the analysis of PCBA schematic drawing in [12, chp. 18 and 20] and the need to include even further information like design history and intentions [29].

6. A General Methodology

This section presents a methodology, divided in four phases, with which to build patch relationships (Section 4). As argued, these patches allow to overcome the mismatch between ontological constraints and domain entities (Section 3).

Given an ontology, we call *ontological property* (or *quality*) a property used to characterize categories in the ontology. We write *basic property* (*quality*) for the properties (qualities) that are primitive in the language of the ontology. Although a basic property can be positive or negative, here we simplify the general case by assuming that categories are characterized by conjunctions of positive properties only. Without loss of generality, we also assume that the basic properties are mutually independent, i.e., none can be defined via a boolean formula in which only other basic properties occur. Given a property P, a *core property* of P is any basic property needed to explicitly define P in the ontology language. (Note that the notion of core property is well defined due to the mutual independence of basic properties.) Let $P_1, P_2, , P_3$ be the basic properties of an ontology and let P be a property which is explicitly defined by a boolean combination of P_1 and P_2 only. Then, P_1 and P_2 (but not P_3) are core properties of P. For instance, if 'begin prince' is formally defined by the conjunction of the basic properties 'being male' and 'being child of a monarch', then the latter two are the only core properties of 'begin prince' in this language. The term *attribute* will be reserved to indicate properties of domain entities. Finally, as discussed at the end of Section 3 we assume that the language of $\mathcal{O}$ has enough (or has been properly extended with) properties to informally model the attributes of the domain entities.

In exemplifying the methodology we will be at times imprecise; the aim is to keep the presentation simple without discussing all the peculiarities of the given ontologies. For instance, we do not list all the relevant categories and, in particular, use only a few of those that classify a PCBA in ISO 15926-2. Also, for practical reasons in the examples we will use first order logic. From the formal viewpoint just a few features, common to most ontological languages [19], are important, namely, existential quantification, conjunction and implication.

Recall that our methodology aims to extend the given ontology by introducing new formal constraints. We rephrase the main properties of the sought extension as follows:

(i) the extended system should be compliant, logically consistent and ontologically coherent with the original ontology or standard

(ii) the company's data already accessible from the ontology must remain accessible from the extended system

(iii) the extended system must allow to correctly access all the data the company considers necessary for the kind of entities at stake

The methodology applies to one category at time, it can be iterated to add more (independent) categories and consists of the following phases:

Phase I) *From attributes to properties*;

Phase II) *Category specialization*;

Phase III) *Patch formula and patch relation*;

Phase IV) *Strenghtening the patch relation*.

Let EC be an *extra-ontological category*, i.e., a category not in $\mathcal{O}$ and let $S_{\mathcal{O}}$ be the set of basic properties in the language of $\mathcal{O}$. For simplicity, we will call EC an *external category*.

6.1. Phase I

The first phase consists in the analysis of the external category EC aimed to isolate a set of properties in the ontology language. The properties should collectively describe (although informally) the necessary conditions for an entity to belong to EC.

(I) *From attributes to properties*

Aim 1: Identification of the minimal set, call it $\mathrm{Basic}_{\mathsf{EC}}$, of basic properties in the language of $\mathcal{O}$ such that they can model (perhaps only informally) any attribute necessary for an entity to belong to EC.

Aim 2: Construction of two formulas out of $\mathrm{Basic}_{\mathsf{EC}}$: one formula, called the $\mathcal{C}$-component, is the conjunction of predicates (whose core properties belong to $\mathrm{Basic}_{\mathsf{EC}}$) that characterize categories in $\mathcal{O}$; the other, called the $\mathcal{R}$-component, is the conjunction of the remaining properties of $\mathrm{Basic}_{\mathsf{EC}}$, if any. The $\mathcal{C}$-component includes only the most specific predicates, i.e., those of the most specialized categories that can be identified with properties from $\mathrm{Basic}_{\mathsf{EC}}$.

This phase is completed with the construction of the following informal equivalence: $a \in \mathsf{EC} \ \cong$ for each category C_i in the $\mathcal{C}$-component there is an entity $a_{C_i} \in C$ in $\mathcal{O}$ which is (perhaps only informally) related to a, and each of the properties P in the $\mathcal{R}$-component is satisfied by at least one of these a_{C_i}'s. The choice of the a_{C_i}'s is determined by the ontology engineer and the relationship between a and each a_{C_i}'s can be stated informally.

EXAMPLE 1: PCBA IN ISO 15926-2

Let EC be the category of PCBA as modeled by the database (DB) of a given company. An element of PCBA is characterized by an identifier and the necessary attributes stated in the DB: the PCBA's time of production, size of the board, base material etc. (Section 3). These attributes characterize properties of some category in $\mathcal{O}$: ARRANGED_INDIVIDUAL (AR) since the DB indicates that the PCBA has components; MATERIAL_PHYSICAL_OBJECT (MO) since the DB includes base material and size of board; and FUNCTIONAL_PHYSICAL_OBJECT (FO) because of attributes like quality test and compatibility.

Let us assume that the set $\mathrm{Basic}_{\mathrm{PCBA}}$ in DB is given by the properties of categories AR, MO, FO plus relations T_p, S_b, F_c, where T_p stands for 'time of production', S_b for 'size of the board' and F_c for 'has compatibility function' (Figure 4). We write $AR(x)$ to mean "x is a member of category AR", $T_p(x, y)$ to mean "x was produced

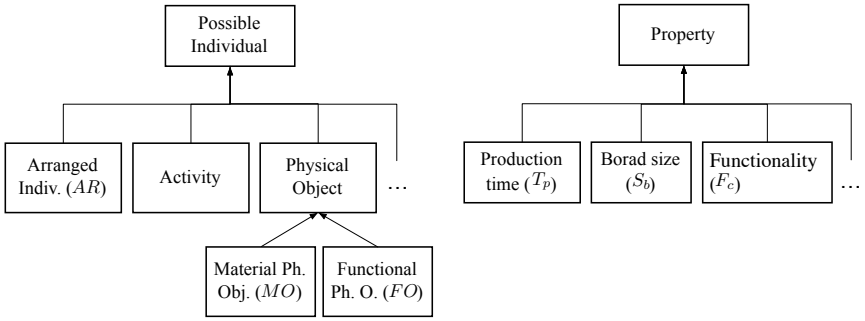

Figure 2: ISO 15926-2 taxonomy (adapted) as used in the PCBA example.

during y", and so on.[5] Informally, expression 'x is a PCBA product' is associated in the language of ISO 15926-2 to formula:

$$\exists y, z, w\ (AR(x) \wedge MO(x) \wedge FO(x) \wedge T_p(x, y) \wedge S_b(x, z) \wedge F_c(x, w))$$

which is the (normalized) conjunction of the $\mathcal{C}$-component, namely $AR(x) \wedge MO(x) \wedge FO(x)$, and the $\mathcal{R}$-component, namely $\exists y, z, w\ (T_p(x, y) \wedge S_b(x, z) \wedge F_c(x, w))$. Note that we use the same x in the whole formula to mark its informal role of "unifying element", thus the formula is not satisfiable in ISO 15926-2: two properties might hold only for different choices of x. For example, T_p requires x to be the production process of the pcba while F_c might be limited to processes starting with the acquirement of some functionality. Also, the variable x has very different readings in the external category and in the ontology: it is an object in PCBA and a process in ISO 15926-2. The whole problem arises since to model the properties of the object x in ISO 15926-2 one should identify the object x with different processes: the object's production process for T_p, the object's whole life for S_b, and the process spanning period in which its functional compatibility satisfies F_c. $\square_{(to\ be\ cont'd)}$

EXAMPLE 2: PCBA DRAWING IN DOLCE

Let EC be the category of PCBA's schematic drawings, call it SD, in the database (DB) of a company. As in the previous case, an element of SD is characterized by an identifier and the associated attributes in the DB: type of support, the author (an individual, a team or a full department), the legend of symbols etc. To model the relationship between identifier (drawing) and author (agent), not present in DOLCE, we use a new relation 'x authored by y', written $Au(x, y)$, such that $Au(x, y)$ implies that x is an artifact ($AF(x)$) and y an agent ($APO(y)$, Figure 3. (AF is is defined in [9].) Let $Sh(x, y)$ stand for "x has shape y" and $Tc(x, y)$ for "x has topic y"; the other basic properties identified in the analysis correspond to categories AF and IE of DOLCE, where $IE(x)$ stands for "x is an information entity". Thus, we have

[5] As discussed earlier, properties like T_p, S_b and F_c might not be properties of the initial language (ISO 15926-2 in this case) but specializations of properties in it.

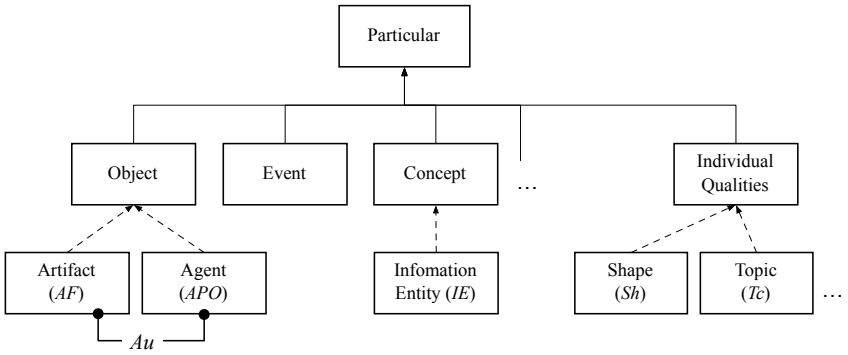

Figure 3: DOLCE taxonomy (adapted) as used in the PCBA drawing example (the new relation Au is also indicated).

$\text{Basic}_{\text{SD}} = \{AF, IE, Sh, Tc, Au\}$. The formula for '$x$ is a PCBA schematic drawing' corresponds to:

$$\exists y, z, w\ (AF(x) \wedge IE(x) \wedge Sh(x, y) \wedge Tc(x, z) \wedge Au(x, w)).$$

The formal expression thus obtained is not satisfiable in DOLCE since categories AF and IE are disjoint. $\square_{(to\ be\ cont'd)}$

At the end of Phase (I) we have an informal "semantic alignment" between ontological properties and the attributes of EC as well as a pseudo-formal representation of the EC in the language of the ontology. Most aspects of this phase rely on human intervention like the match between an attribute and the related ontological properties, or the verification of the $\mathcal{C}$- and $\mathcal{R}$-components.

6.2. Phase II

Here the ontological properties in the $\mathcal{R}$-component are used to specialize the ontology, if needed. The outcome is the ontology $\mathcal{O}$ possibly extended with new categories obtained as specializations of leaf categories.

(II) *Category specialization*.

Aim: Introduction of a minimal set of subcategories in $\mathcal{O}$ as specializations of existing leaf categories such that: if a category C has characteristic property P_C (occurring in the $\mathcal{C}$-component) and is compatible to property P_a in the $\mathcal{R}$-component, then a new category C_a is added as subcategory of C in $\mathcal{O}$. The elements of C_a are the elements of C that satisfy P_a.

EXAMPLE 1: PCBA IN ISO 15926-2 ($cont'd$)
Our previous analysis led to associate 'x is a PCBA product' to this existential formula: $\exists y, z, w\ (AR(x) \wedge MO(x) \wedge FO(x) \wedge T_p(x, y) \wedge S_b(x, z) \wedge F_c(x, w))$. We now specialize the categories using T_p, S_b, F_c. Le us introduce AR_T, MO_T and FO_T

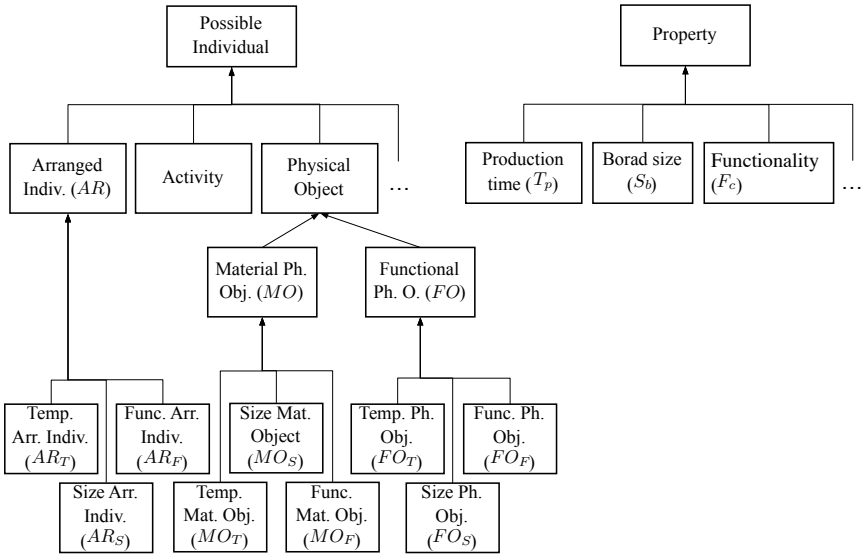

Figure 4: ISO 15926-2 taxonomy as expanded in phase (II).

as specializations of AR, MO and FO so that, e.g., AR_T collects all the arranged individuals spanning just the production process (usually the initial part of the whole process). Analogously, MO_T and FO_T collect the production part of any process classified by MO and FO, respectively. Similarly, we obtain specializations relative to size and functionality: AR_S, MO_S, FO_S, AR_F, MO_F and FO_F, see Figure 4. Note that we specialize all the categories since they are all compatible with the properties in the $\mathcal{R}$-component. More generally, one specializes only those categories that can have elements satisfying the property at stake.[6] $\square_{(to\ be\ cont'd)}$

EXAMPLE 2: PCBA DRAWING IN DOLCE ($cont'd$)
In Phase (I) expression 'x is a PCBA schematic drawing' was associated to formula: $\exists y, z, w\ (AF(x) \wedge IE(x) \wedge Sh(x, y) \wedge Tc(x, z) \wedge Au(x, w))$. In DOLCE all elements of AF have shape (since inherited by the constituting physical objects) and turn out to have a creator while all elements of IE have topic, all properties are satisfied by each element in at least one of these categories. It follows that for SD no category of DOLCE needs to be specialized further. (This conclusion is due to our restriction to general properties of SD. Had we included, e.g., the artifact material (electronic, paper, microfilm etc.), we would have to introduce corresponding specializations since DOLCE does not subdivide artifacts along these features.) $\square_{(to\ be\ cont'd)}$

[6] If a property in the $\mathcal{R}$-component is not compatible with any category in the $\mathcal{C}$-component then there was a misjudgment in the choice of the categories in phase (I) since all basic properties are satisfiable in $\mathcal{O}$.

6.3. Phase III

This phase introduces new relationships, called *patches*, between the external category EC and the ontological categories of $\mathcal{O}$. The term 'patch' is chosen to suggest that the relationships make justice of the necessary attributes of EC's elements by patching together ontological categories and basic properties.

(III) *Patch formula and patch relation*

Aim: Introduction of a formula (the *patch formula*) in the mixed domain of quantification given by that of $\mathcal{O}$ and that of the external category. The formula has as arguments a variable ranging on the category EC (universally quantified argument) and one variable for each specialized category of phase II as well as for each category of phase I that has not been specialized (existentially quantified arguments). Given an element a in EC, the formula forces the existence of entities om $\mathcal{O}$ related to a. The main relationship in the formula, called the *patch relation*, is at the core of the connection between EC and $\mathcal{O}$.

EXAMPLE 1: PCBA IN ISO 15926-2 ($cont'd$)
In phase (II) we added categories: AR_T, MO_T and FO_T; AR_S, MO_S and FO_S; AR_F, MO_F and FO_F. These categories cover all properties identified in phase (I). The patch formula states that for each entity in EC, there exist entities a_i, m_i, f_i such that: $a_1 \in AR_T$, $a_2 \in AR_S$, $a_3 \in AR_F$, $m_1 \in MO_T$, $m_2 \in MO_S$, $m_3 \in MO_F$, $f_1 \in FO_T$, $f_2 \in FO_S$ and $f_3 \in FO_F$. Formally, the patch is a $\forall\exists$ implication formula with a new relation $R_{\text{PCBA}-ISO}$, called the patch relation (Figure 5), such that ($\vec{y}$ stands for y_1, y_2, y_3):

$$\forall \mathtt{x} \exists \vec{y}, \vec{z}, \vec{w}\ (\text{PCBA}(\mathtt{x}) \rightarrow R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y}, \vec{z}, \vec{w})),$$

where $R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y}, \vec{z}, \vec{w})$ implies $AR_T(y_1) \wedge AR_S(y_2) \wedge AR_F(y_3) \wedge MO_T(z_1) \wedge MO_S(z_2) \wedge MO_F(z_3) \wedge FO_T(w_1) \wedge FO_S(w_2) \wedge FO_F(w_3)$. $\square_{(to\ be\ cont'd)}$

EXAMPLE 2: PCBA DRAWING IN DOLCE ($cont'd$)
In phase (II) no specialization of DOLCE categories has been introduced, we thus proceed with the categories from phase (I). Let $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z)$ be the patch relation (Figure 6); the patch formula for the SD category is:

$$\forall \mathtt{x} \exists y, z\ (SD(\mathtt{x}) \rightarrow R_{SD-\text{DOLCE}}(\mathtt{x}, y, z)),$$

where $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z)$ implies $AF(y) \wedge IE(z)$. $\square_{(to\ be\ cont'd)}$

Finally, recall that the elements of the category EC are not compatible with $\mathcal{O}$, thus the patch relationship does not become part of the language of $\mathcal{O}$.[7] It plays the role of a bridge between the view of the company (as cast in the database) and the view of the ontology $\mathcal{O}$.

[7] In the patch formulas x is written with a different font to highlight that it ranges outside $\mathcal{O}$'s domain.

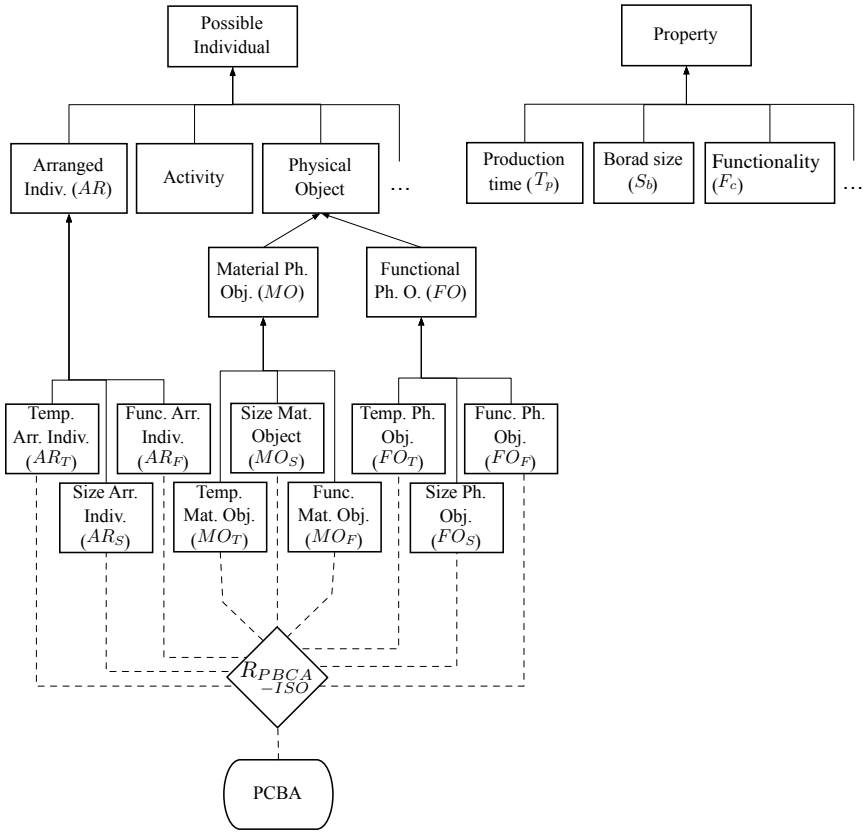

Figure 5: Expanded ISO 15926-2 taxonomy and the patch relation $R_{\text{PCBA}-ISO}$.

6.4. Phase IV

This phase aims to add further constraints to the patch relation to constrain its interpretation. The strengthening is obtained by formalizing relationships analyzed at phase (I).

(IV) *Strengthening of the patch relation*
Aim: Introduction of a set of formulas to constrain the interpretation of the patch relation. While the patch formula forces the existence of entities in the selected ontological categories, the new formulas of this phase aim to bind those entities to bear the right interdependences.

EXAMPLE 1: PCBA IN ISO 15926-2 ($cont'd$)
From $\mathcal{O}$'s perspective a PCBA is at the same time a process in the categories AR, MO and FO. Since we specialized these categories to distinguish process according

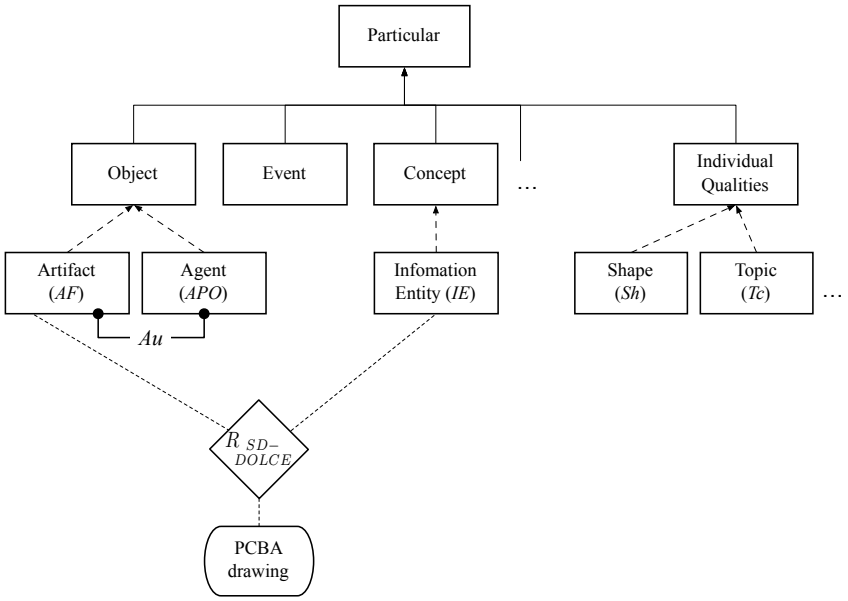

Figure 6: DOLCE taxonomy and the patch relation for PCBA drawings.

to the different properties for a PCBA, we can state that the elements in the specialized categories must be the same:

- $R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y}, \vec{z}, \vec{w}) \rightarrow \bigwedge_{1 \leqslant i \leqslant 3} y_i = z_i = w_i)$
  (anything related to a PCBA is in $AR \cap M \cap F$)

Since the elements of ISO 15926-2 identified by the formula can have a shorter life than the PCBA itself (e.g. the PCBA passes the quality test only some time after it is produced and it may acquire new qualifications even years later when declared compatible with newer products), one can force all the elements associated via the $R_{\text{PCBA}-ISO}$ to have physical continuity across time by adding, e.g., formula:

- $R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y}, \vec{z}, \vec{w}) \wedge R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y'}, \vec{z'}, \vec{w'}) \rightarrow \exists \vec{u}\, (\bigwedge_{i \leqslant 3} AR(u_i) \wedge \bigwedge_{i \leqslant 3} MO(u_i) \wedge \bigwedge_{i \leqslant 3} FO(u_i) \wedge \text{TempPart}(y_i, u_i) \wedge \text{TempPart}(z_i, u_i) \wedge \text{TempPart}(w_i, u_i))$
  (anything related by $R_{\text{PCBA}-ISO}$ to the same PCBA is part of a unique process)

Stricter constraints can also be considered, e.g.:

- $R_{\text{PCBA}-ISO}(\mathtt{x}, \vec{y}, \vec{z}, \vec{w}) \rightarrow \bigwedge_{i \leqslant 3} (\text{TempPart}(y_i, z_i) \wedge \text{TempPart}(w_i, z_i))$.

One can also add continuity constraints on single components (or the PCBA board itself) to prevent a PCBA to undergo specific replacements, if this is the view of the company. Analogously, one can constrain features like quality checks, to be verified after something else has been accomplished: component check, production completion,

delivery (for externally produced elements), etc. These constraints, if considered during the analysis at phase (I), can all be captured in ISO 15926-2 via the $R_{\text{PCBA}-ISO}$ relation. $\square_{(end)}$

EXAMPLE 2: PCBA DRAWING IN DOLCE ($cont'd$)
From $\mathcal{O}$'s perspective a single PCBA schematic drawing is both a support entity (physical or electronic) and a content entity, thus we constrain each argument in AF to be related to a unique element of IE and vice versa by, for instance:

- $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z) \wedge R_{SD-\text{DOLCE}}(\mathtt{x}, y', z')) \rightarrow y = y' \wedge z = z'$
  (a schematic drawing is related to a unique support and content)

Also, the global content of a support is unique but the content itself does not identify the support where it is cast (since the same content can be reproduced in different supports):

- $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z) \wedge R_{SD-\text{DOLCE}}(\mathtt{x}', y, z') \rightarrow \mathtt{x} = \mathtt{x}' \wedge z = z'$
  (a support of a PCBA drawing is related to a unique global content)
- $\exists \mathtt{x}, \mathtt{x}', y, y', z\ (R_{SD-\text{DOLCE}}(\mathtt{x}, y, z) \wedge R_{SD-\text{DOLCE}}(\mathtt{x}', y', z) \wedge \mathtt{x} \neq \mathtt{x}')$
  (content identifies neither a PCBA drawing nor a support)

$\square_{(end)}$

Since all attributes necessary to belong to EC have been translated at phase (I), all DB data relative to the category EC are accessible in the integrated system as required by condition (iii) of page 13. Also, the ontology maintains its integrity since no structural change has been introduced beside allowed category specialization (e.g., the creation of new leafs like AR_T) and the introduction of new properties or relations (e.g., Au) with their characterization in the original language of the ontology itself. Also, logical consistency follows since patch formulas are not part of the ontology, and ontological coherence is guaranteed provided the analysis at phase (I) is correctly carried out. Thus condition (i) is satisfied as well. The part of the DB directly accessible from the ontology and the original ontology language have not changed, thus condition (ii). Finally, patches have an important indirect impact on the ontology: they introduce new relations among ontological entities (but not necessarily between categories). When we write $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z) \wedge R_{SD-\text{DOLCE}}(\mathtt{x}, y', z') \rightarrow y = y' \wedge z = z'$, we do not change the ontology but force it to be populated with a specific drawing and a specific content object; and when we state $R_{SD-\text{DOLCE}}(\mathtt{x}, y, z) \wedge R_{SD-\text{DOLCE}}(\mathtt{x}', y, z') \rightarrow \mathtt{x} = \mathtt{x}' \wedge z = z'$ we practically add to the ontology a constraint of type: $R'(y, z) \wedge R'(y, z') \rightarrow z = z'$ where R' is the projection of $R_{SD-\text{DOLCE}}$ on its second and third arguments. This type of constraints can be taken as characteristic of a subcategory or simply limited to some elements in $\mathcal{O}$ depending on how one wants to strengthen the resulting system.

7. Conclusions

We have focused on a general problem for the application of ontologies and semantic-based standards to industry: the exploitation of information regarding categories not

compatible with the adopted ontological taxonomy. We verified that existing solutions in areas like linguistics, logic and ontology do not help to solve this problem. We then described a new methodology which, given an ontology $\mathcal{O}$, is based on the analysis of the extra-ontological (external) category in terms of the basic ontological properties. The approach relies on the idea of dependence relations which are well understood in ontology research but their use for ontology integration is still uncommon. The main advantage of our approach is that it leaves the general ontology unaltered while making available all information the company has about the domain entities.

We gave principles for the methodology and verified that three important requirements are satisfied by its application, namely, (i) The system with the patch relationship between the ontology and the extra-ontological category is compliant, logically consistent and ontologically coherent with the company's view; (ii) The company's data are all accessible; and (iii) The company's perspective is not modified in the integration.

Future work aims to clarify three important aspects for the implementation of the methodology. First, the definition of patch relations for two or more extra-ontological but interdependent categories (leading to a general methodology for database and ontology integration). In fact, the methodology described here can be applied to integrate only a single category at a time with the further restrictions that to consider another extra-ontological category, the latter must be unrelated to extra-ontological categories previously patched. (To generalize this point, we need a deeper analysis of the dependence relations.) Second, the definition of general inference rules to fully exploit the patch formulas and the patch relationships. Third, the integration of the patch methodology with the notion of view in database literature. This latter notion has received much attention but remains at the logical level. The logical structure of the formulas for views and patches is very close; the main difference between these approaches is in the analysis carried out to build them, and thus in their content. We believe it is possible to integrate these analyses with the goal of merging views and patches; the merging would give a general methodology that takes advantage of all the relevant aspects: syntax, semantics and ontological analysis.

8. Acknolwedgements

This work has been initiated during the Marie-Curie IRSES EuJoint project (n. 247503) and developed within the Gecko project (part of the Flagship Project "Factory of the Future") funded by the MIUR - Ministero dell'Istruzione, dell'Università e della Ricerca.

[1] N. A. Anjum, J. Harding, R. Young, and K. Case. Mediation of foundation ontology based knowledge sources. *Computers in Industry*, 63(5):433 – 442, 2012.

[2] N. Asher and A. Lascarides. *Logics of Conversation*. Studies in Natural Language Processing. CUP, 2003.

[3] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge Univ. Press, 2003.

[4] R. Batres, M. West, D. Leal, D. Price, K. Masaki, Y. Shimada, T. Fuchino, and Y. Naka. An upper ontology based on ISO 15926. *Computers & Chemical Engineering*, 31(5–6):519 – 534, 2007.

[5] L. Bellatreche, N. X. Dung, G. Pierra, and D. Hondjack. Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases. *Computers in Industry*, 57(8):711–724, 2006.

[6] S. Borgo, M. Carrara, P. Garbacz, and P. E. Vermaas. A formalization of functions as operations on flows. *Journal of Computing and Information Science in Engineering*, 11(3), 2011.

[7] S. Borgo and P. Leitão. Foundations for a Core Ontology of Manufacturing. In S. R. Kishore R., Ramesh R., editor, *Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems*, volume 14 of *Integrated Series in Information Systems*, pages 751–776. Springer, 2007.

[8] S. Borgo and C. Masolo. Foundational Choices in DOLCE. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, International handbooks on information systems, pages 361–381. Springer Verlag, 2nd edition, 2009.

[9] S. Borgo and L. Vieu. Artifacts in Formal Ontology. In Anthonie Meijers, editor, *Handbook of the Philosophy of the Technological Sciences. Technology and Engineering Sciences*, vol. 9, pg. 273–307. Elsevier, 2009.

[10] V. Chulvi and R. Vidal. B-cube, behavioural modelling of technical artefacts. *Computers in Industry*, 64(1):68 – 79, 2013.

[11] J. Conesa and A. Olivé. A method for pruning ontologies in the development of conceptual schemas of information systems. In S. Spaccapietra, P. Atzeni, W. W. Chu, T. Catarci, and K. Sycara, editors, *Journal on Data Semantics V*, volume 3870 of *Lecture Notes in Computer Science*, pages 64–90. Springer Berlin Heidelberg, 2006.

[12] C. F. Coombs, editor. *Printed circuits handbook*. McGraw-Hill New York, 6th edition, 2008.

[13] F. Correia. Ontological dependence. *Philosophy Compass*, 3(5):1013–1032, 2008.

[14] V. Ebrahimipour, K. Rezaie, and S. Shokravi. An ontology approach to support FMEA studies. *Expert Systems with Applications*, 37(1):671 – 677, 2010.

[15] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[16] K. Fine. Ontological dependence. *Proceedings of the Aristotelian Society*, 95:269–90, 1994.

[17] G. Flouris, D. Manakanatas, H. Kondylakis, D. Plexousakis and G. Antoniou. Ontology change: classification and survey. *The Knowledge Engineering Review*, 23(2):117–152, 2008

[18] E. Folmer, P. Oude Luttighuis, and J. Hillegersberg. Do semantic standards lack quality? a survey among 34 semantic standards. *Electronic Markets*, 21(2):99–111, 2011.

[19] V. Fortineau, T. Paviot, and S. Lamouri. Improving the interoperability of industrial information systems with description logic-based models—the state of the art. *Computers in Industry*, 64(4):363 – 375, 2013.

[20] P. Garbacz, S. Borgo, M. Carrara, and P. E. Vermaas. Two ontology-driven formalisations of functions and their comparison. *Journal of Engineering Design*, 22(11-12):733–764, 2011.

[21] D. M. Giménez, M. Vegetti, H. P. Leone, and G. P. Henning. Product ontology: Defining product-related concepts for logistics planning activities. *Computers in Industry*, 59(2):231–241, 2008.

[22] T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6):907–928, 1995.

[23] T. Grubic and I.-S. Fan. Supply chain ontology: Review, analysis and synthesis. *Computers in Industry*, 61(8):776 – 786, 2010. <ce:title>Semantic Web Computing in Industry</ce:title>.

[24] N. Guarino and C. Welty. Towards a methodology for ontology-based model engineering. In *ECOOP-2000 Workshop on Model Engineering*, Cannes, France, 2000.

[25] N. Guarino and C. Welty. An overview on ontoclean. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 201–220. Springer Verlag, 2nd edition, 2009.

[26] ISO 15926-2. *ISO-15926:2003 Integration of lifecycle data for process plant including oil and gas production facilities: Part 2 – Data model*. International Organization for Standardization, 2003.

[27] C. M. Keet and A. Artale. Representing and reasoning over a taxonomy of part–whole relations. *Applied Ontology*, 3:91–110, 2008.

[28] B. C. Kim, H. Teijgeler, D. Mun, and S. Han. Integration of distributed plant lifecycle data using ISO 15926 and web services. *Annals of Nuclear Energy*, 38(11):2309 – 2318, 2011.

[29] J. Kim, M. J. Pratt, R. G. Iyer, and R. D. Sriram. Standardized data exchange of {CAD} models with design intent. *Computer-Aided Design*, 40(7):760 – 777, 2008. <ce:title>Current State and Future of Product Data Technologies (PDT)</ce:title>.

[30] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider. The wonderweb library of foundational ontologies. Deliverable 17, EU WonderWeb Project,
http://wonderweb.man.ac.uk/deliverables.shtml, 2002.

[31] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider. The wonderweb library of foundational ontologies. Deliverable 17, EU WonderWeb Project,
http://wonderweb.man.ac.uk/deliverables.shtml, 2002.

[32] N. T. Nguyen. A method for ontology conflict resolution and integration on relation level. *Cybernetics and Systems*, 38(8):781–797, 2007.

[33] N. F. Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.*, 33(4):65–70, 2004.

[34] F. S. Parreiras, S. Staab, S. Schenk, and A. Winter. Model driven specification of ontology translations. In Springer-Verlag, editor, *Proceedings of the 27th International Conference on Conceptual Modeling*, volume Lecture Notes In Computer Science; Vol. 5231, pages 484 – 497, 2008.

[35] A. Poggi, D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *Journal of Data Semantics*, X:133–173, 2008.

[36] G. Priest. *TOWARDS NON-BEING, the logic and metaphysics of intentionality*. Oxfod University Press, New York (USA), paperback edition, 2005.

[37] W. V. O. Quine. *From a logical point of view*. Harvard University Press, Cambridge, MA, 1953.

[38] W. V. O. Quine. Quantifiers and propositional attitudes. *The Journal of Philosophy*, 53:177–187, 1956.

[39] S. Ram and J. Park. Semantic conflict resolution ontology (scrol): An ontology for detecting and resolving data and schema-level semantic conflicts. *IEEE Transactions on Knowledge and Data Engineering*, 16(2):189 – 202, February 2004.

[40] A. Rector. Modularisation of domain ontologies implemented in description logics and related formalisms including owl. In *Proceedings of the International Conference on Knowledge Capture*, pages 121–128. ACM Press, 2003.

[41] P. Simons. *Parts: a Study in Ontology*. Clarendon Press, Oxford, Oxford, 1987.

[42] S. SK, P. MK, and O. LJ. Toward the use of an upper ontology for u.s. government and u.s. military domains: An evaluation. Technical Report Technical Report MTR 04B0000063, The MITRE Corporation, 2004.

[43] S. Staab and R. Studer. *Handbook on Ontologies*. International handbooks on information systems. Springer Verlag, Berlin (DE), 2nd edition, 2009.

[44] V. C. Storey, R. Chiang, and G. L. Chen. Ontology creation: Extraction of domain knowledge from web documents. In Springer-Verlag, editor, *Conceptual Modeling – ER 2005*, volume 3716, pages 256–269, 2005.

[45] A. L. Thomasson. Methods of categorization. In A. Varzi and L. Vieu, editors, *Formal Ontology in Information Systems (FOIS04)*, pages 3–16, Turin, 2004. IOS Press.

[46] L. Vieu and M. Aurnague. Part-of relations, functionality and dependence. In *The Categorization of Spatial Entities in Language and Cognition*, volume 20 of *Human Cognitive Processing*, pages 307–336. John Benjamins Publishing Company, 2007.

[47] J. V́olker, D. Vrandečić, Y. Sure, and A. Hotho. Aeon - an approach to the automatic evaluation of ontologies. *Applied Ontology*, 3(1-2):41–62, 2008.

[48] R. Volz, R. Studer, A. Maedche, and B. Lauser. Pruning-based identification of domain ontologies. *Universal Comput. Sci.*, 9(6):520–529, 2003.

[49] D. C. Wimalasuriya and D. Dou. Using multiple ontologies in information extraction. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 235–244, New York, NY, USA, 2009. ACM.

[50] C. Zanni-Merk, D. Cavallucci, and F. Rousselot. Use of formal ontologies as a foundation for inventive design studies. *Computers in Industry*, 62(3):323 – 336, 2011.