# A HIERARCHICAL PRODUCT-PROPERTY MODEL TO SUPPORT PRODUCT CLASSIFICATION AND MANAGE STRUCTURAL AND PLANNING DATA

Diego M. Giménez[1], Gabriela P. Henning[1], Horacio P. Leone[2]

[1] INTEC (UNL-CONICET), Güemes 3450, Santa Fe S3000GLN, Argentina
[2] INGAR (UTN-CONICET), Avellaneda 3657, Santa Fe S3002GJC, Argentina
dgimenez@fiq.unl.edu.ar, ghenning@intec.unl.edu.ar, hleone@conicet-santafe.gov.ar

**Abstract.** Mass customization is one of the main challenges that managers face since it results in a proliferation of product data within the various organizational areas of an enterprise and across different enterprises. Effective solutions to this problem have resorted to generic bills of materials and to the grouping of *product variants* into *product families*, thus improving data management and sharing. However, issues like product family identification and formation, as well as data aggregation have not been dealt with by this type of approach. This contribution addresses these challenges and proposes a hierarchical data model based on the concepts of *variant*, *variant set* and *family*. It allows managing huge amounts of structural and non-structural information in a systematic way, with minimum replication. Besides, it proposes an unambiguous criterion, based on the *properties* of *variants*, for identifying *families* and *variant sets*. Finally, the approach can explicitly handle aggregated data which is intrinsic to generic concepts like *families* and *variant sets*. A case study is analyzed to illustrate the representation capabilities of this approach.

**Keywords:** Product Data Model, Multiple Levels of Abstraction, Product Properties, Product Classification.

## 1 Introduction

Today's enterprises are forced to offer products which fulfill individual customer needs. Because of the growing mass customization, industrial environments are characterized by a large product variety. Therefore, an efficient treatment of product data is required in order to deal with the huge amount of product-related information that is managed daily within an enterprise and exchanged across different enterprises.

Several authors have proposed alternative solutions to the management of massive product data, based on the concepts of Product Family and Generic Bill of Materials [1,2]. In this direction, Giménez et al. [3] presented a novel product ontology named PRONTO. The core of this approach is a three-level *abstraction hierarchy*, where the lower level concerns physical products, which are handled through the concept of *variant*. The upper level abstracts a population of *variants* with some similar characteristics and is managed through the concept of *family*. The intermediate level, handled through the concept of *variant set*, represents a set of *variants* with many similar characteristics. Thus, the *abstraction hierarchy* is constructed by grouping *variants* into *variant sets* and them into *families* according to their "similarity". This abstraction approach is helpful to share and not replicating common information. It is also very useful when carrying out planning activities at different levels (strategic, tactical and operational) since aggregated information can be generated along the hierarchy. Nevertheless, some issues, like product classification have not been treated in this line of research.

In this paper an unambiguous criterion for product classification is formalized. Despite many works in the specialized literature addressed issues regarding classification [4], the novelty of this work lies in that the proposed classification mechanism is compatible with the approaches for managing common structural and non-structural information as well as generating product data of different granularity. Specifically, the classification criterion focuses on the *properties* that are defined in order to describe the attributes of *variants*.

The paper is organized as follows. The concepts, relations and conditions on which the proposal is based are explained in Section 2. Section 3 presents a case study to illustrate the representation capabilities of the proposed approach. To conclude, some final remarks are presented in Section 4.

## 2 Proposed Approach

### 2.1 Product Classification

The proposed representation relies on an *abstraction hierarchy* having three levels, which are represented in the class diagram of Fig. 1 by the **Variant**, **VariantSet** and **Family** classes.

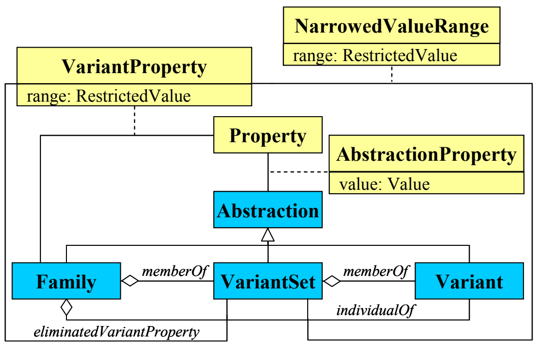



**Fig. 1.** The three-level abstraction hierarchy.

According to the classification criterion to be formalized in this section, a *family* is defined as a set of actual products or *variants* with several common *properties*, having values within a specified range. Similarly, a *variant set* is defined as a subset of *variants*, within a given *family*, which have the same *properties* and whose values are within a given range, included in the one defined for the *family*. Thus, the *variant set* notion can be seen as a subfamily concept.

Regarding the *abstraction hierarchy*, any concept (**Abstraction**) is of just one type (*family*, *variant set* or *variant*). If $\boldsymbol{A}$ denotes the entire set of *abstractions* and $\boldsymbol{I}$, $\boldsymbol{J}$, and $\boldsymbol{K}$ a partition of $\boldsymbol{A}$ representing the subsets of *families*, *variant sets* and *variants*, respectively; then, conditions specified in (1) must be satisfied.

$$(a)\ \boldsymbol{A} = \boldsymbol{I} \cup \boldsymbol{J} \cup \boldsymbol{K}\ ;(b)\ \boldsymbol{I} \cap \boldsymbol{J} = \{\ \};(c)\ \boldsymbol{I} \cap \boldsymbol{K} = \{\ \};(d)\ \boldsymbol{J} \cap \boldsymbol{K} = \{\ \} \qquad (1)$$

In turn, each product instance or *variant* is a member of only one *variant set* and each *variant set* is a member of just one *family*. If $\boldsymbol{K}_j$ and $\boldsymbol{J}_i$ denote the set of members (*variants*) of *variant set j* and the set of members (*variant sets*) of *family i*, respectively, conditions prescribed in (2) and (3) are imposed.

$$(a)\ \forall k \in \boldsymbol{K}\ \exists\ j \in \boldsymbol{J} : k \in \boldsymbol{K}_j\ ;(b)\ \forall k \in \boldsymbol{K}, k \in \boldsymbol{K}_j \wedge k \in \boldsymbol{K}_{j'} \Leftrightarrow j = j' \qquad (2)$$

$$(a)\ \forall j \in \boldsymbol{J}\ \exists\ i \in \boldsymbol{I} : j \in \boldsymbol{J}_i\ ;(b)\ \forall j \in \boldsymbol{J}, j \in \boldsymbol{J}_i \wedge j \in \boldsymbol{J}_{i'} \Leftrightarrow i = i' \qquad (3)$$

Consequently, each *variant* belongs to only one *family*, as clauses in (4) prescribe.

$$(a) \forall k \in \boldsymbol{K} : k \in \boldsymbol{K}_j \wedge j \in \boldsymbol{J}_i \Rightarrow k \in \boldsymbol{K}_i ;(b) \forall k \in \boldsymbol{K}, k \in \boldsymbol{K}_i \wedge k \in \boldsymbol{K}_{i'} \Leftrightarrow i = i' \qquad (4)$$

where $\boldsymbol{K}_i$ denotes the set of individuals of *family i*.

*Properties* play an essential role in product classification mechanisms. Two kinds of *properties* are proposed: (i) The ones associated with the individuals of a *family*, which are represented by the **Variant Property** concept (Fig. 1). This notion allows specifying for each *family* the *properties* of its population, as well as their ranges of possible values. Likewise, at the *variant set* level, the subset of *properties* shared by all its members is specified by removing those *variant properties,* associated with the corresponding *family,* that do not belong to the members of the *variant set* (*eliminated VariantProperty*). Though *variant properties* are specified at the *family* level, they generally assume values at the level of specific instances or *variants*. (ii) Properties that are particular or intrinsic to a generic concept, like *family* or *variant set*; hence, they are assigned values at the level of their definition. This notion is modeled by the **AbstractionProperty** association (Fig. 1). Property concepts are formalized in (5),

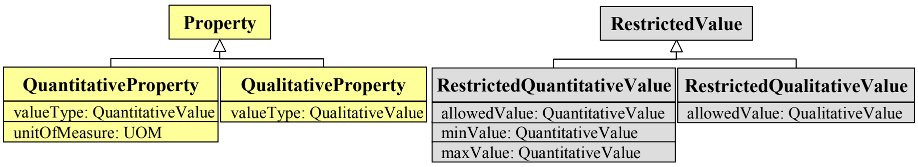
$$(a)\ \forall j \in \boldsymbol{J} : j \in \boldsymbol{J_i} \Rightarrow \boldsymbol{P_j^K} \subseteq \boldsymbol{P_i^K}\ ;(b)\ \forall k \in \boldsymbol{K} : k \in \boldsymbol{K_j} \Rightarrow \boldsymbol{P_k} = \boldsymbol{P_j^K} \qquad (5)$$

where $\boldsymbol{P}_i^K$ is the set of *variant properties* associated with *family i*, $\boldsymbol{P}_j^K$ the subset of *variant properties* shared by the members of *variant set j*, and $\boldsymbol{P}_k$ the set of *properties* of *variant k*. Moreover, the value range of a given *variant property,* specified at the *family* level, can be reduced when defining a *variant set* (**NarrowedValueRange**).

*Properties* are classified into *qualitative* and *quantitative* ones. In both cases the value type is specified, and for *quantitative properties*, the unit of measure must be indicated (see Fig. 2a).

The value range of *variant properties* is represented by the class **RestrictedValue** which, in turn, is specialized into quantitative and qualitative categories (see Fig. 2b).

Within these subclasses a discrete set of allowed values can be specified. It is also possible to define a continuous range for quantitative values.



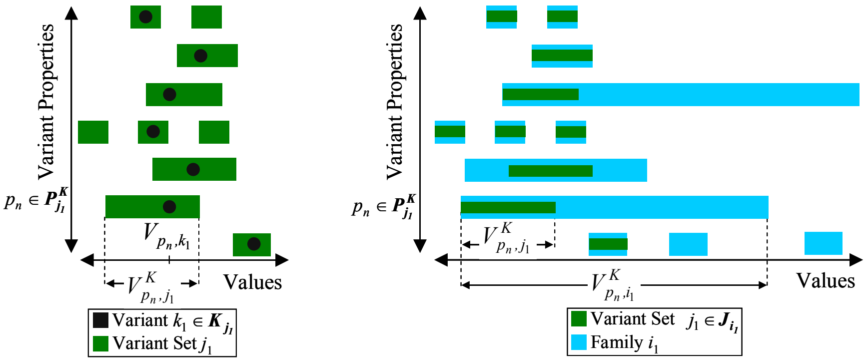

(a) **Property** categories (b) **Value Range** categories

**Fig. 2. Property** and **Restricted Value** specializations.

As mentioned before, if a particular *variant* is a member of a given *variant set*, then each *variant property* must assume values belonging to the range specified by the *variant set*, as prescribed in (6).

$$\forall k \in \boldsymbol{K} : k \in \boldsymbol{K}_{\boldsymbol{j}} \Rightarrow \forall p \in \boldsymbol{P}_{\boldsymbol{k}}, \boldsymbol{V}_{\boldsymbol{p,k}} \subseteq \boldsymbol{V}^{\boldsymbol{K}}_{\boldsymbol{p,j}} \tag{6}$$

where $\boldsymbol{V}_{p,k}$ is the set of values that *property p* assumes for *variant k* and $\boldsymbol{V}_{p,j}{}^{K}$ in the range of possible values for *property p* set by *variant set j*. Fig. 3a conceptualizes this idea.




(a) *Variant* classification (b) *Variant set* classification

**Fig. 3.** *Variant* and *Variant Set* classification conceptual notions.

Similarly, if a given *variant set* is a member of a certain *family*, then, for each *variant property* pertaining to such *variant set*, the range of possible values must be included within the range fixed by the *family*. This specification is formalized in (7).

$$\forall j \in \boldsymbol{J} : j \in \boldsymbol{J}_{\boldsymbol{i}} \Rightarrow \forall p \in \boldsymbol{P}^{\boldsymbol{K}}_{\boldsymbol{j}}, \boldsymbol{V}^{\boldsymbol{K}}_{\boldsymbol{p,j}} \subseteq \boldsymbol{V}^{\boldsymbol{K}}_{\boldsymbol{p,i}} \tag{7}$$

where $\boldsymbol{V}_{p,i}{}^{K}$ is the value range corresponding to *property p*, which is defined by *family i*. This notion is conceptually shown in Fig. 3b.

## 2.2 Product Unambiguous Definition

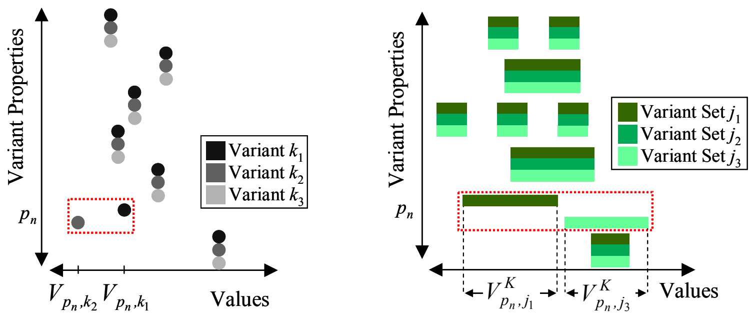
One basic assumption of the proposed model is that each product concept must be unambiguously identified. This would allow implementing proper classification mechanisms. Thus, at the lower level of the hierarchy it is implied that all *variants* must be different. In the context of this approach two *variants* are considered distinct if either their sets of *properties* are different or there exists at least one *property* which assumes dissimilar values for each of these *variants*, as formalized in (8).

$$\forall\, k, k' \in \boldsymbol{K},\;\; k \neq k' \Leftrightarrow \boldsymbol{P_k} \neq \boldsymbol{P_{k'}} \vee \exists\, p \in \boldsymbol{P_k} : \boldsymbol{V_{p,k}} \neq \boldsymbol{V_{p,k'}} \tag{8}$$

Fig. 4(a) conceptualizes various *properties* and their corresponding values for three different *variants*. As it can be seen, $\boldsymbol{P}_{k_1}=\boldsymbol{P}_{k_2}\neq\boldsymbol{P}_{k_3}$ and $\boldsymbol{V}_{p_n,k_1}\neq\boldsymbol{V}_{p_n,k_2}$, therefore $k_1\neq k_2\neq k_3$. Likewise, all *variant sets* must be strictly different. This occurs if either they specify distinct subsets of *variant properties* or there exists at least one associated *variant property* for which the intersection between their corresponding value ranges is empty. This notion is formally specified in (9).

$$\forall\, j, j' \in \boldsymbol{J},\;\; j \neq j' \Leftrightarrow \boldsymbol{P_j^K} \neq \boldsymbol{P_{j'}^K} \vee \exists\, p \in \boldsymbol{P_j^K} : \boldsymbol{V_{p,j}^K} \cap \boldsymbol{V_{p,j'}^K} = \{\,\} \tag{9}$$

Fig. 4(b) conceptualizes several *variant properties* and their corresponding value ranges for three different *variants sets*. As it can be seen, $\boldsymbol{P}_{j_1}^{K}=\boldsymbol{P}_{j_3}^{K}\neq\boldsymbol{P}_{j_2}^{K}$ and $\boldsymbol{V}_{p_n,j_1}^{K}\neq\boldsymbol{V}_{p_n,j_3}^{K}$, therefore $j_1\neq j_2\neq j_3$.




(a) *Variant* uniqueness (b) *Variant Set* uniqueness

**Fig. 4.** *Variant* and *Variant Set* uniqueness schematic representations.

Finally, *families* should be strictly different. This occurs if they specify distinct sets of *variant properties* or there exists at least one associated *variant property* for which the intersection of their corresponding value ranges is empty, as prescribed in (10).

$$\forall\, i, i' \in \boldsymbol{I},\;\; i \neq i' \Leftrightarrow \boldsymbol{L_i^K} \neq \boldsymbol{L_{i'}^K} \vee \exists\, p \in \boldsymbol{P_i^K} : \boldsymbol{V_{p,i}^K} \cap \boldsymbol{V_{p,i'}^K} = \{\,\} \tag{10}$$

Fig. 5 conceptually shows various *variant properties* and their corresponding value ranges for three different *families*. As it can be seen, $\boldsymbol{P}_{j_1}^{K}=\boldsymbol{P}_{j_3}^{K}\neq\boldsymbol{P}_{j_2}^{K}$ and $\boldsymbol{V}_{p_n,j_1}^{K}\neq\boldsymbol{V}_{p_n,j_3}^{K}$, therefore $j_1\neq j_2\neq j_3$. These assumptions assure an unambiguous classification criterion.

In consequence, each *variant* should be a member of a unique *variant set* and each *variant set* should be a member of a unique *family*. Therefore, each *variant* would belong to a unique *family*.

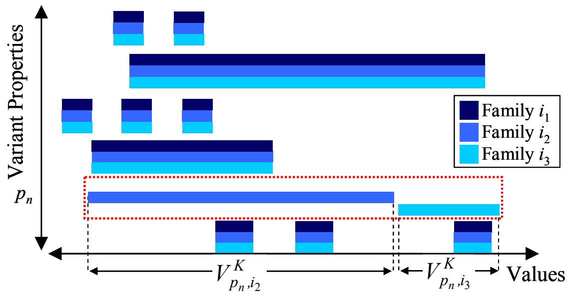



**Fig. 5.** *Family* uniqueness schematic representation.

### 2.3 Product Structure

Another essential challenge of product modeling is the representation of product structures. Regarding this issue, *families* are classified into *compound* and *simple families*. In the first case, *compound families* can be decomposed into other *families* (a set of "parts" can be identified). On the other hand, *simple families* cannot be further decomposed. Along the same line of reasoning, *variant sets* and *variants* are classified into *compound* and *simple variant sets*, and *compound* and *simple variants*, respectively. These notions are formally stated in (11) and (12).

$$(a)\ \boldsymbol{I} = \boldsymbol{I}^{\boldsymbol{C}} \cup \boldsymbol{I}^{\boldsymbol{S}}\ ;(b)\ \boldsymbol{J} = \boldsymbol{J}^{\boldsymbol{C}} \cup \boldsymbol{J}^{\boldsymbol{S}}\ ;(c)\ \boldsymbol{K} = \boldsymbol{K}^{\boldsymbol{C}} \cup \boldsymbol{K}^{\boldsymbol{S}} \tag{11}$$

$$(a)\ \boldsymbol{I}^{\boldsymbol{C}} \cap \boldsymbol{I}^{\boldsymbol{S}} = \{\ \};(b)\ \boldsymbol{J}^{\boldsymbol{C}} \cap \boldsymbol{J}^{\boldsymbol{S}} = \{\ \};(c)\ \boldsymbol{K}^{\boldsymbol{C}} \cap \boldsymbol{K}^{\boldsymbol{S}} = \{\ \} \tag{12}$$

where $\mathbf{I}^C/\mathbf{J}^C/\mathbf{K}^C$ is the subset of compound families/variant sets/variants and $\mathbf{I}^S/\mathbf{J}^S/\mathbf{K}^S$ is the subset of simple families/variant sets/variants.

To keep model consistency, it is assumed that low-level *compound/simple abstractions* are members of high-level *compound/simple abstractions*. These assumptions are prescribed in (13) to (15).

$$(a)\ \forall k \in \boldsymbol{K}^{\boldsymbol{C}} : k \in \boldsymbol{K}_j \Rightarrow j \in \boldsymbol{J}^{\boldsymbol{C}}\ ;(b)\ \forall k \in \boldsymbol{K}^{\boldsymbol{S}} : k \in \boldsymbol{K}_j \Rightarrow j \in \boldsymbol{J}^{\boldsymbol{S}} \tag{13}$$

$$(a)\ \forall j \in \boldsymbol{J}^{\boldsymbol{C}} : j \in \boldsymbol{J}_i \Rightarrow i \in \boldsymbol{I}^{\boldsymbol{C}}\ ;(b)\ \forall j \in \boldsymbol{J}^{\boldsymbol{S}} : j \in \boldsymbol{J}_i \Rightarrow i \in \boldsymbol{I}^{\boldsymbol{S}} \tag{14}$$

$$(a)\ \forall k \in \boldsymbol{K}^{\boldsymbol{C}}\ \exists i \in \boldsymbol{I}^{\boldsymbol{C}} : k \in \boldsymbol{K}_{\boldsymbol{i}}\ ;(b)\ \forall k \in \boldsymbol{K}^{\boldsymbol{S}}\ \exists i \in \boldsymbol{I}^{\boldsymbol{S}} : k \in \boldsymbol{K}_{\boldsymbol{i}} \tag{15}$$

According to the proposal of Giménez et al. [3], *generic structures* are defined for *compound families*, which in turn can be modified by *compound variant sets* in order to allow the construction of particular BOMs for *compound variants*. Specifically, one or more *generic structures* are associated with each *compound family*. Then, each *variant set* specifies one *generic structure* of the corresponding *family* and from this particular one it derives the *structure* shared by all its members. Finally, actual BOMs (at the *variant* level) are obtained from the *structure* defined at *variant set* level.

*Generic structures* are classified into *composition* and *decomposition structures* depending on whether the *compound family* is composed of *generic components* (*families*) or it is decomposed into *generic derivatives* (*families*), as indicated in (16).

$$(a)\ \forall i \in \boldsymbol{I}^{\boldsymbol{C}}\ \exists s \in \boldsymbol{S} : s \in \boldsymbol{S}_{i}\ ;(b)\ \boldsymbol{S} = \boldsymbol{S}^{\boldsymbol{C}} \cup \boldsymbol{S}^{\boldsymbol{D}}\ ;(c)\ \boldsymbol{S}^{\boldsymbol{C}} \cap \boldsymbol{S}^{\boldsymbol{D}} = \{\ \} \tag{16}$$

where $\boldsymbol{S}$ is the set of *structures*, $\boldsymbol{S}_i$ is the subset of *structures* associated with *family* $i$, and $\boldsymbol{S}^C/\boldsymbol{S}^D$ the subset of *composition/decomposition structures*.

The class diagram shown in Fig. 6 illustrates the specialization of the *family* concept (**Family)** into *compound* and *simple family* (**CFamily** and **SFamily**, respectively), the specialization of the *generic structure* concept (**Structure**) into *composition* and *decomposition structures* (**CStructure** and **DStructure**, respectively), and the definition of *generic structures* through *structural relations* (**StructuralRelation**), which are classified into *composition* and *decomposition structural relations* (**CStructuralRelation** and **CStructuralRelation**, respectively).

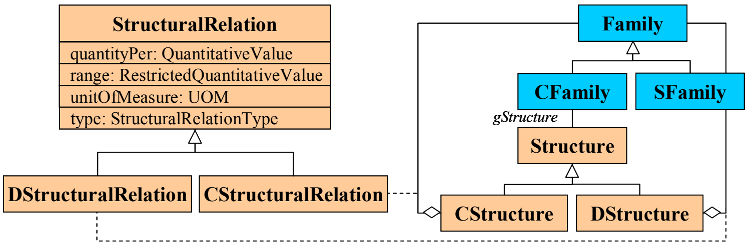



**Fig. 6.** *Generic structure* representation.

A *structural relation* is established between the *generic structure* and the corresponding generic components/derivatives (*compound* or *simple families*). This relation provides information about the quantity of the generic component/derivative required/obtained per unit of *compound family*. In addition, the range of possible values for the quantity mentioned above, its unit of measure, and the relation type are also defined.

Three types of *structural relations* are adopted: *mandatory*, *optional* and *selective*. The chosen type determines whether a given *structural relation* can be removed from a *generic structure* by a *variant set*. Thus, when it is *mandatory*, the relation must exist; if it is *optional*, it can be eliminated; and when it is *selective*, only one relation of this type must be chosen (the other ones must be removed). See clauses (17)-(18).

$$(a)\ \forall s \in \boldsymbol{S}^{\boldsymbol{C}}\ \exists i \in \boldsymbol{I} : i \in \boldsymbol{I}_s^{\boldsymbol{GC}}\ ;(b)\ \boldsymbol{I}_s^{\boldsymbol{GC}} = \boldsymbol{I}_s^{\boldsymbol{GCm}} \cup \boldsymbol{I}_s^{\boldsymbol{GCo}} \cup \boldsymbol{I}_s^{\boldsymbol{GCs}} \tag{17}$$

$$(a)\ \forall s \in \boldsymbol{S}^{\boldsymbol{D}}\ \exists i \in \boldsymbol{I} : i \in \boldsymbol{I}_s^{\boldsymbol{GD}}\ ;(b)\ \boldsymbol{I}_s^{\boldsymbol{GD}} = \boldsymbol{I}_s^{\boldsymbol{GDm}} \cup \boldsymbol{I}_s^{\boldsymbol{GDo}} \cup \boldsymbol{I}_s^{\boldsymbol{GDs}} \tag{18}$$

where $\boldsymbol{I}_s^{GC}/\boldsymbol{I}_s^{GD}$ is the set of generic components/derivatives for *structure* $s$, $\boldsymbol{I}_s^{GCm}/\boldsymbol{I}_s^{GCo}/\boldsymbol{I}_s^{GCs}$ the subset of *mandatory/optional/selective* generic components for *structure s* and $I_s^{GDm}/I_s^{GDo}/I_s^{GDs}$ the subset of *mandatory/optional/selective* generic derivatives for *structure s*. As mentioned before, a given *compound variant set* specifies one (and only one) of the *generic structures* associated with the *family* of which it is member, being able to eliminate some *structural relations*, but just those that are not *mandatory*. See clauses (19)-(23) representing these concepts.

$$(a)\forall j \in \boldsymbol{J^C} : j \in \boldsymbol{J_i}\ \exists s \in \boldsymbol{S_i} : j \in \boldsymbol{J_s^C} ;(b)\forall j \in \boldsymbol{J^C}, j \in \boldsymbol{J_s^C} \wedge j \in \boldsymbol{J_{s'}^C} \Leftrightarrow s = s' \quad (19)$$

$$\forall j \in \boldsymbol{J_s^C} : s \in \boldsymbol{S^C}\ \exists i \in \boldsymbol{I_s^{GC}} : i \in \boldsymbol{I_j^{GC}} \quad (20)$$

$$\forall j \in \boldsymbol{J_s^C} : s \in \boldsymbol{S^C} \Rightarrow \boldsymbol{I_j^{GC}} \subseteq \boldsymbol{I_s^{GC}} \wedge \boldsymbol{I_s^{GCm}} \subseteq \boldsymbol{I_j^{GC}} \quad (21)$$

$$\forall j \in \boldsymbol{J_s^C} : s \in \boldsymbol{S^D}\ \exists i \in \boldsymbol{I_s^{GD}} : i \in \boldsymbol{I_j^{GD}} \quad (22)$$

$$\forall j \in \boldsymbol{J_s^C} : s \in \boldsymbol{S^D} \Rightarrow \boldsymbol{I_j^{GD}} \subseteq \boldsymbol{I_s^{GD}} \wedge \boldsymbol{I_s^{GDm}} \subseteq \boldsymbol{I_j^{GD}} \quad (23)$$

where $\boldsymbol{J_s^C}$ is the subset of *compound variant sets* whose *structure* derives from *generic structure s* and $\boldsymbol{I_j^{GC}}/\boldsymbol{I_j^{GD}}$ is the set of generic components/derivatives (*families*) from which the *variant sets* taking part in the *structure* of *variant set j* are selected.

Thus, a particular *variant set* must be selected for each non-eliminated generic component/derivative. In other words, for each *family* assuming the role of generic component/derivative in a non-eliminated *structural relation*, a *variant set* being member of such a *family* must be specified, as prescribed in (24).

$$(a)\ \forall i \in \boldsymbol{I_j^{GC}}\ \exists j' \in \boldsymbol{J_i} : j' \in \boldsymbol{J_j^{GC}}\ ;(b)\ \forall i \in \boldsymbol{I_j^{GD}}\ \exists j' \in \boldsymbol{J_i} : j' \in \boldsymbol{J_j^{GD}} \quad (24)$$

where $\boldsymbol{J_j^{GC}}/\boldsymbol{J_j^{GD}}$ is the set of components/derivatives of *variant set j*. Fig. 7 depicts the representation of the *structure* of a *compound variant set*.

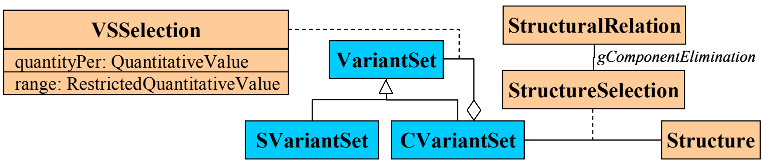



**Fig. 7.** Compound variant set structure representation.

When a *variant set selection* is carried out, a new “quantity per” value and a new range of possible values can be specified. The condition to be satisfied is that the new range must be included within the range stipulated by the associated *structural relation*.

Finally, each *compound variant* adopts the *structure* defined by the *variant set* of which it is a member of, and specifies a particular member (*variant*) of each *variant set* assuming the generic component/derivative rol in such a *structure.* See (25)-(26).

$$k \in \boldsymbol{K_j} : j \in \boldsymbol{J_s^C} \wedge s \in \boldsymbol{S^C} \Rightarrow \forall j' \in \boldsymbol{J_j^{GC}}\ \exists k' \in \boldsymbol{K_{j'}} : k' \in \boldsymbol{K_k^C} \quad (25)$$

$$k \in \boldsymbol{K}_j \wedge j \in \boldsymbol{J}_s^C \wedge s \in \boldsymbol{S}^D \Rightarrow \forall\, j' \in \boldsymbol{J}_j^{GD}\ \exists\, k' \in \boldsymbol{K}_{j'} : k' \in \boldsymbol{K}_k^D \qquad (26)$$

where $\boldsymbol{K}_k^C/\boldsymbol{K}_k^D$ is the set of components/derivatives of *variant* $k$. Fig. 8 shows the single-level BOM representation corresponding to a *compound variant*.

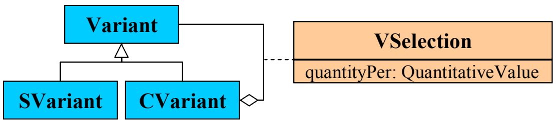



**Fig. 8.** *Variant* BOM representation.

## 3 Case Study

In this section, the proposed approach is employed to represent the data associated with the set of cookware products illustrated in Fig. 9.

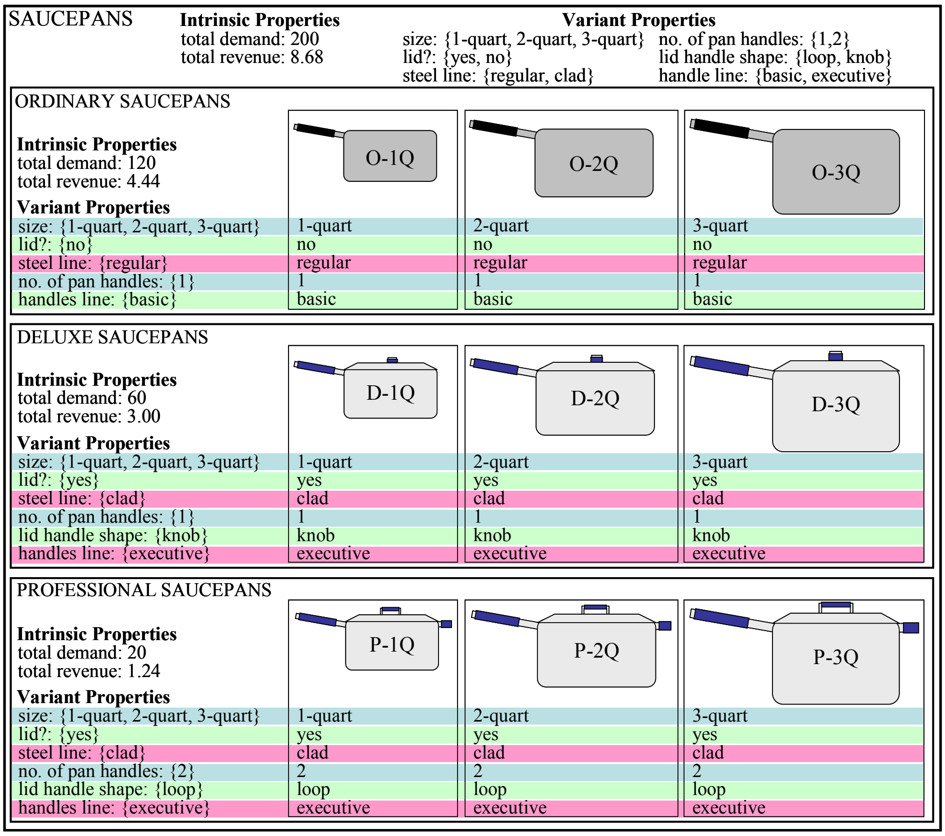



**Fig. 9.** Set of products considered in the case study.

As it can be seen, the set of products corresponds to the *family* of saucepans. Three *variant sets* were identified: ordinary, deluxe and professional saucepans; each one grouping three *variants*. The first *variant set* represents the economical line of saucepans. They are characterized by having only one pan handle, not having a lid and being manufactured with medium-quality materials. The second *variant set* represents the intermediate product line. In this case, saucepans are characterized by having one pan handle, a lid with knob handle and being manufactured with high-quality materials. The last *variant set* represents the most complete line of products, which are recognized by having two pan handles, a lid with loop handle, apart from being manufactured with high-quality materials. *Variants* within a given *variant set* differ only in size. The three standardized sizes of saucepans are “one-quarter”, “two-quarter” and “three-quarter”. The *abstraction hierarchy* regarding this case study is depicted in Fig. 10.

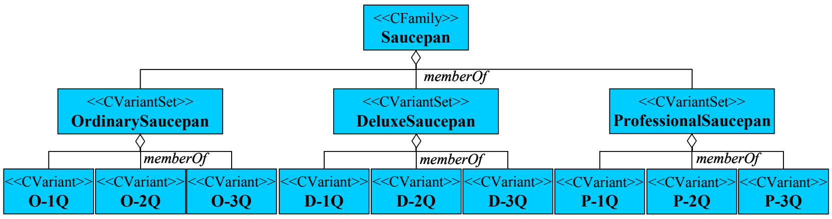



**Fig. 10.** *Abstraction hierarchy* associated with the case study.

Clearly, *family*, *variant sets*, and *variants* are *compound abstractions,* since they are composed of other *abstractions* representing subassemblies. Fig. 9 also presents some of the *properties* associated with the different levels of abstraction. Two *intrinsic properties* were exemplified for the *family* of saucepans (*total demand* and *total revenue*) and six *variant properties* were included in such a *family* (*size*, *lid?*, *steel line*, *no. of pan handles*, *lid handle shape* and *handle line*). Besides, the same *intrinsic properties* were defined for all *variant sets* (i.e. *total demand* and *total revenue*).

In relation to *variant properties*, the *lid handle shape* one was eliminated from ordinary saucepans since they have no lids. Property values and value ranges are also shown in Fig. 9. Some examples of the concepts and relations presented in Figs. 1 to 3 are given in Figs. 11 and 12. Basically, Fig. 11 shows the definition of a specific *variant property* by a given *family* and the elimination of such a *variant property* by a particular *variant set*. It is also shown the narrowing of its value range. In turn, Fig. 12 illustrates examples of *intrinsic properties* associated with *product abstractions*.

From Fig. 9 it can be seen that the ranges specified for *variant properties* at the *variant set* level are comprised within the range stipulated for each *variant property* at the corresponding *family* level. Alike, values of *variant properties* specified at the *variant* level are comprised within the range established by the corresponding *variant set*.

On the other hand, it is verified that all *variant sets* are strictly different. Ordinary saucepans are different from deluxe and professional ones because the sets of *variant properties* are distinct. Despite possessing the same *variant properties*, deluxe and professional saucepans are also different because the value ranges defined for some *variant properties* have no common elements (values). For example, the number of pan handles is fixed to one for deluxe saucepans and to two for professional ones. Moreover, all *variants* are dissimilar since, within each *variant set,* they vary in size, as mentioned before.

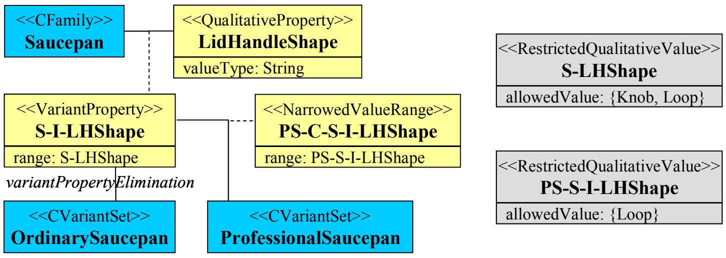



**Fig. 11.** *Variant property* definition.

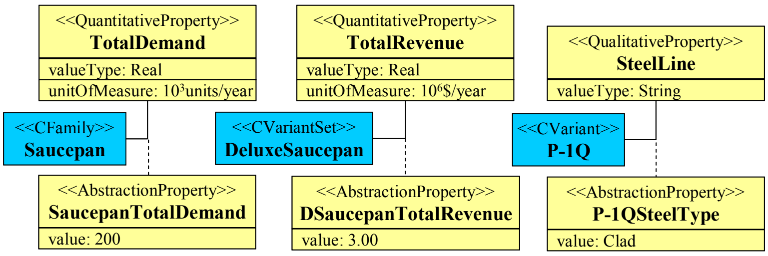



**Fig. 12.** Representation of *abstraction properties*.

Regarding *product structures*, the *generic composition structure* associated with the *family* is represented in Fig. 13. Saucepans are generically composed of a pan assembly (*mandatory*) and a lid assembly (*optional*). In both cases the "quantity per" is exactly equal to 1.

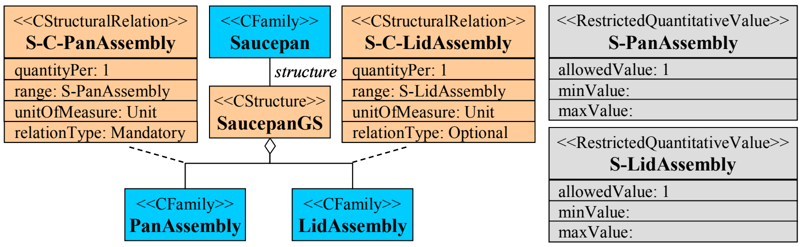
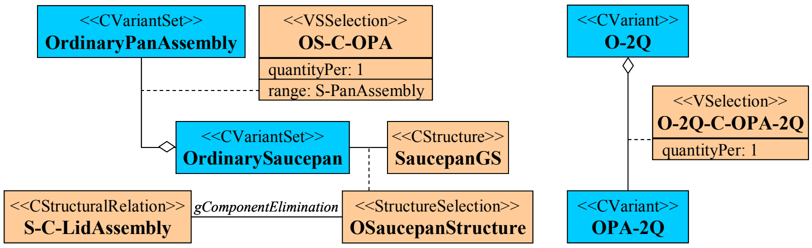



**Fig. 13.** Saucepans´ *generic structure*.

An example of a *variant set structure* is shown in Fig. 14(a). In this case, ordinary saucepans are composed of only ordinary pan assemblies. The *family* of lid assemblies was eliminated from the *generic structure*. In turn, the set of ordinary pan assemblies was selected as a generic component. An example of a *variant* single-level BOM is depicted in Fig. 14(b). As it is shown, a two-quarter ordinary saucepan is composed of a two-quarter ordinary pan assembly.




(a) Ordinary saucepans *structure* (b) 2-Q ordinary saucepan BOM

**Fig. 14.** Examples of the *Variant Set structure* and *Variant* BOM generation concepts.

## 4 Final Remarks

In this paper, a novel model for product data management is presented. The proposal is based on a three-level abstraction hierarchy. It differs from similar approaches because an unambiguous criterion to classify product concepts along the hierarchy is defined. At the same time, it provides foundations to handle data (dis)aggregation processes in a systematic way. Moreover, the proposed model attempts to offer more expressiveness, flexibility and reuse of information at the different levels of abstraction.

Regarding model validation, a prototype of a Distributed Product Data Management (DPDM) system that supports the classification criterion described in Section 2 is currently under development. Several case studies of different complexity are being addressed in order to validate the conceptual model and the classification procedure as well as to evaluate its practical applicability. Preliminary results show that many of the complexities associated with the management of massive product data can be effectively tackled by implementing this novel hierarchical product-property model.

**Acknowledgments.** This work has been supported by CONICET, UTN, and UNL.

## References

1. De Lit, P., Danloy, J., Delchambre, A., Henrioud, J-M.: An Assembly-Oriented Product Family Representation for Integrated Design. IEEE Transactions on Robotics and Automation, 19, 75–88 (2003)
2. Du, X., Jiao, J., Tseng, M.M.: Architecture of Product Family: Fundamentals and Methodology. Concurrent Engineering: Research and Application, 9, 309–325 (2001)
3. Giménez, D.M., Vegetti, M., Henning, G.P., Leone, H.P.: PRoduct ONTOlogy: Defining product-related concepts for logistics planning activities. Computers in Industry, 59, 231–241 (2008)
4. Yan, W., Chen, S-H., Huang, Y., Mi, W.: A data-mining approach for product conceptualization in a web-based architecture. Computers in Industry, 60, 21–34 (2009)