CONFIGURATION MANAGEMENT BY DOLCE UPPER-LEVEL ONTOLOGY

Dragoș ILIESCU[1], Manuela-Roxana DIJMĂRESCU[1] and Marian GHEORGHE[1]

ABSTRACT: The product development process requires progress of the processes and activities within the Product Lifecycle Management by adding improvements to its components. This paperwork presents the results achieved in the field of Configuration Management by the use of the product, versions and variants informational models, ontological foundation axioms conferring soundness to the obtained results. Clarification on the identity of product master identification represents the achievements of this work, contributions being made to Configuration Management concept. Functionality to be fulfilled by the object that the end user expects – the product, and qualities to be inherited by it, define the two viewpoints presented in the work: the client and the manufacturer viewpoints product definition, along with versions and variants identification criteria.

KEY WORDS: product, product development, configuration management, upper-level ontology, DOLCE.

1 INTRODUCTION

A continuous development in the field of assisted product development has been noted since early 90's. The results could be observed with multiple connections in Computer Aided Design, Computer Aided Manufacturing, Computer Aided Engineering, Product Data Management or Product Life-cycle Management concepts and applications. A contradiction between standardisation and customisation of technical solutions soon claimed to be negotiated.

The necessary solutions came in the form of Configuration Management described in terms of recommendation by ISO 10007 "Quality management systems – Guidelines for configuration management" (ISO, 2005) or by the initiative described in ISO 10303 "Standard for the Exchange of Product model data" (STEP) (Ungerer, 2002). Even if the clarification has generated useful effects, there is still a need for further clarifications. These aspects are solved by the use of upper-level ontologies generating effects upon definitions that operate in the field.

[1]Universitatea POLITEHNICA din Bucuresti, Facultatea de Ingineria și Managementul Sistemelor Tehnologice, Splaiul Independentei 313, 060042 Bucuresti, Romania

E-mail: dragosiliescu2005@gmail.com; manuela-d@live.com; marian.gheorghe@upb.ro

2 DOLCE UPPER-LEVEL ONTOLOGY

Several ontologies, having the domains of discourses with particular or with universal entities, are available today (Mascardi, 2007).

The ontology "Descriptive Ontology for Linguistic and Cognitive Engineering" (DOLCE), has the domain of discourse defined by the particular entities. This domain considers the following components: endurants, perdurants, qualities and abstract entities.

Endurants and perdurants are mainly defined according with their existence in a space-time dimension (Gangemi, 2002). Endurants are explained as they wholly exist in a space dimension while the perdurants evolve in a time or a space-time dimension. There are strong relationships between endurants and perdurants due to their connections with the space-time region. Endurants participate in perdurants, both can be composed by other endurants, respectively by other perdurants, and both can assign qualities. Endurants are defined in a 3D space dimension; perdurants are defined in a 4D space-time dimension. Several basic relations are considered within the logical foundation of DOLCE (Masolo, 2008): parthood, constitution, participation, quality and quale. A specific structure showing the decomposition of DOLCE is depicted in Figure 1. Endurants (*ED*) can be in form of: physical endurant (*PED*), non-physical endurant (*NPED*) and mereological (arbitrary) sum (*AS*). *PED* are in form of: physical object (*POB*), amount of matter (*AM*) and feature (*F*). *NPED* are in form of: non-physical object (*NPOB*) which is in form of:

mental object (*MOB*) and social object (*SOB*). Perdurants (*PD*) are in form of: process (*PRO*) and event (*EV*). These entities can assign achievement (*ACH*), accomplishment (*ACC*) and state (*ST*).

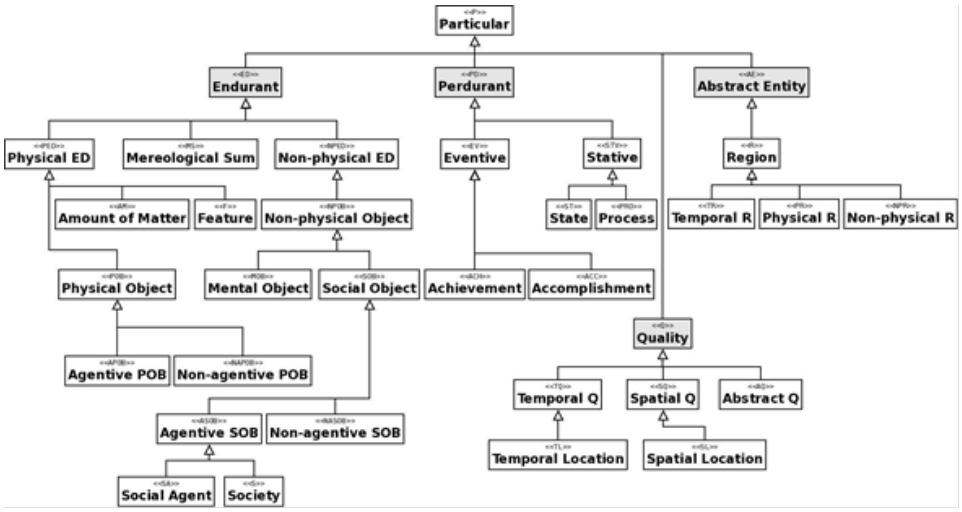

Figure 1. DOLCE categories, *Source: (Masolo, 2008)*

3 PRODUCT AND PROCESS DOLCE DEFINITION

It is important to define the entities of products and processes in order to make possible further constructions. ISO 9001 standard defines the process as "*a set of inter-related activities that transforms inputs into outputs*". The definition of the product is "*product is the result of a process*" (ISO, 2008). Such definitions, even correct ones, are not able to emphasize all the aspects that the process entity or the product entity can have. STEP initiative gives the identification definition of a product as a structure composed on: product, version, variant and view (Ungerer, 2002), a structure named in STEP terminology as master identification. This initiative introduces the entity of product in relation with the version entity. The product entity is seen as that collection of all the versions that product can have in a given moment in time, but the primary goal is to offer an informational model of product and version or variant entities. By the use of the view entity, the integrated model technique is approached too. Nevertheless, the above definitions express the product meaning from the manufacturer viewpoint, but a client or end-user perspective is still not being revealed. The DOLCE ontology can bring needed clarification by the use of ground ontological relations (Borgo, 2010).

3.1 Client viewpoint

Considering a client's request (*Req*) for a specific object (*POB*) to fulfil a specific need or function (*Func*), the relation (1) describes the client viewpoint (Borgo, 2010):

$$Req(f,y,v,w) \rightarrow Func(f) \wedge POB(y) \wedge InFlow(v) \wedge OutFlow(w) \quad (1)$$

Considering the ISO 9001 process definition, terms of *InFlow(v)* and *OutFlow(w)* can be described (equation 2) in a different context (Borgo, 2010):

$$PC(InFlow(v),Func(f)) \rightarrow OutFlow(w) \quad (2)$$

The above relation states the product definition from the client viewpoint: participation (*PC*) of inputs (*InFlow*) into the perdurants of *Func* will produce *OutFlow*. This participation, and the relation between the *InFlow* and *OutFlow*, transforms the entity of *POB* into the entity of *APOB* describing also the client (or in some cases the end-user) viewpoint. Both *InFlow* and *OutFlow* can be decomposed as: material (*MFlow*), energy (*EFlow*) and signal (*SFlow*) as shown by equation 3 (Borgo, 2010), (Borgo & Carrara2009). Significance of the eq. (3) is important because the description of the client perspective. It states that the *PRO* can operate upon a material flow, transforming the input materials into the output

material, or it can transfer energy or a signal flow into the output flow. Specific for the client or the end-user perspective is that the perdurants will transform or it will transfer a flow from input to the output.

$$InFlow \vee OutFlow \leftrightarrow Mflow \wedge EFlow \wedge SFlow \quad (3)$$

$$PC(PED(y) \wedge MOB(y) \wedge ASOB(y), PRO(f) \vee EV(f)) \rightarrow POB(r) \quad (4)$$

3.2 Manufacturer viewpoint

Considering the process (*PRO*) and product (*POB*) as its result, the relation (4) expresses the manufacturer viewpoint, i.e., participation of inputs such as: physical endurants, mental objects and agentive social objects into process or event will result into the client expected physical object. Figure 2 depicts the manufacturer viewpoint of process and product definitions.

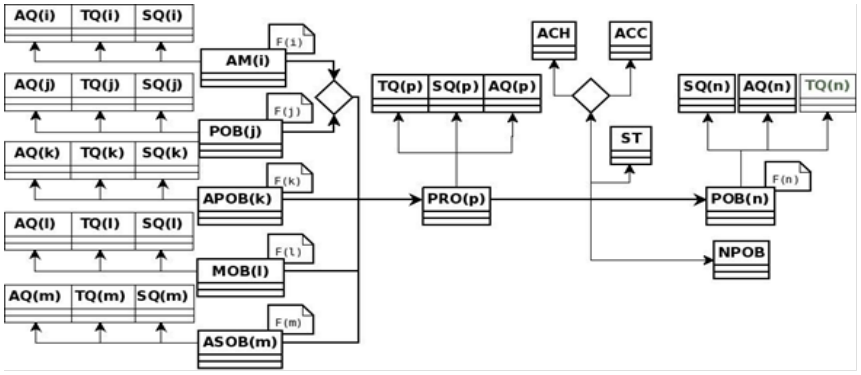

Figure 2. Process and product relationship definition

The InFlow entity is shown as: amount of matter or physical object, agentive physical object, mental object and agentive social object. This classification has the meaning that the process *PRO* creates a new physical object by aggregating the amount of matter *AM*, or it transforms the input physical object into the resulting physical object.

The process use agentive physical object (*APOB*) as facility resources and agentive social object (*ASOB*) as human resources. The process can be measured (defined) by temporal (*TQ*), spatial (*SQ*) and abstract (*AQ*) qualities; the process can be characterized by the entities of achievement (*ACH*), accomplishment (*ACC*) and state (*ST*). The result is obtained in form of *POB* defined by *SQ* and *AQ*. Sometimes it needs also *TQ* due to temporal aspects of the obtained *POB*. The qualities of *SQ(n)*, *AQ(n)* and *TQ(n)* are the expression of client requirements engineered in the form of product qualities. These are direct connected with product development activity, meaning that in the phase of concept development there is a translation from the client perspective to the designer perspective, finally translated into the manufacturer perspective.

Several applicable techniques, Quality Function Deployment being a good example, provides a good contribution in this direction. The involved process should be defined here as in a non-atomic form ($\neg$ *At(Func(f))*. Along with the *POB*, there is a special output from the *PRO* in the form of *NPOB*. This is represented by the amount of data coming out from the *PRO*, describing characteristics of *PRO* and *InFlow*, or interactions among them. *NPOB* is then transformed into information by the usage of specific actions. When enough information is available, the information should be transformed into knowledge that latter will influence the product design adding more value to the product design. Shortly, this is the path of engineering changes and the route of how a new product version is developed. Nevertheless, in the manufacturer viewpoint, the *PRO* creates or transforms the inputs endurants into the output endurants. Differences are noted for the manufacturer viewpoint against the client, or end-user viewpoint where the *Func(f)* transfers or transforms the input flow into the output flow.

4 PRODUCT IDENTITY

The identity of the product, a set of criteria to decide that two endurants are equal or not (Masolo, 2008), (Borgo, 2009), is of high importance. The product can be defined from DOLCE perspective as that object which satisfies the client need, or that object which is defined as result of the assigned qualities aggregation. First definition derives from

relation (1), however, the second definition is expressed bellow (eq. (5)):

$$POB(y) \leftrightarrow I(x,y) \wedge Q(x) \quad (5)$$

Between *Func(f)* and inhere *I(x,y)* is a very strong relationship. It can be stated that *Func(f)* determines the set of qualities inherited by the bearer object. The reverse is also true. The *POB* is therefore defined by the *Func(f)*, but the same *POB* is also defined by aggregation of qualities too. This is expressed in eq. (6).

$$Func(f) \leftrightarrow Agreg\ (I(x,y) \wedge Q(x) \wedge POB(y) \quad (6)$$

There are two entities in debate: function and qualities. The first entity is the one that is defined by the client, usually it is expressed as a verb or an activity along with an artefact, but there is not a formalized form; the second is defined by the product developer as result of some specific activity.

4.1 Function

This entity describes what the client (end user) needs. Often described in a short form, it is the engineering task to define the decomposition of this function (eq. (7)).

$$Func(f) = Agreg(f_1, f_2, f_3 \ldots) \quad (7)$$

A function *Func(f)* represents an aggregation of certain sub functions (eq. (7)) and it is considered that there can be no unity across those sub functions. The necessary decomposition will start from the eq. (1) and by several techniques of product functional design or concept development (Kitsios, 2000), (Zha, 2006), the sub functions can be derived; they are used as *InFlow* for definition of the product structure. After the function decomposition is completed, the next step is to determine the qualities needed by the product. One sub function will derive into one or more qualities, but the interactions will also be under consideration.

4.2 Qualities

The product to fulfil a specified function requires a set of qualities to be specified. This set can be requested (also suggested or proposed) by the client, or can be engineered during product development. There is a strong relationship between the necessary set of qualities and the decomposition of function above. It can be stated that two equal *POBs* show two identical sets of qualities. This is based on the assumption that the relation between quality and bearer is so strong that there is a deterministic relation in-between (Borgo, 2010), (Maurin, 2014):

$$I(t,y_1) \wedge I(t,y_2) \rightarrow y_1 = y_2 \quad (8)$$

4.3 Changes

There are certain changes that can impact the product, the corresponding version, or the variant. Using a non-atomic formulation, consider eq. (6) and (8) that can define the impact on the changes in quality set against the product identity. This change will result in two different entities. That is to say, based on trope theory (Maurin, 2014), two endurants that assign two non-identical set of qualities, resolving different functions, are different by definition. On the other hand, changes not affecting the set of qualities as a whole will determine a new product version. One can formulate this as:

$$Ver(y) = POB(y) \wedge I(Agreg(x),y) \wedge Q(x) \quad (9)$$

Resulting from above, the identity of product version (*Ver*) is dependant with the aggregate of $Q(x) = Agreg(x_i)$, because the term of *POB(y)* is constant. It is possible to modify the *k-term* of $Agreg(x_i)$ without changing the result of qualities aggregation – *Q(x)* is therefore possible. In such a case, there is a new product version, often assimilated with engineering change concept. Other changes can be noted if there is a variation in quale of x_i term, but this will define the product variant.

$$Var(y) = Ver(y) \wedge i(\alpha,x) \wedge q(\alpha) \quad (10)$$

Identity of the product variant is dependant with the instance *i* applied to *x*'s *quale(α)*. That is to say, any change in quale of any assigned quality will result in a new variant. Figure 3 depicts the dependencies across the product, version and variants against the market segment, engineering changes and so on.

5 CASE STUDY: THE HAIR DRYER

In 1911 the hand held hair dryer was invented, as different from the hood hair dryer, and since then a lot of new designs tried to improve this useful appliance. A schema of hand held hair dryer engineering history is depicted in Figure 4. For exemplifying purposes, consider the hair dryer product function *Func(f)*:

Func(f) = "*dry a person's hair in shortest duration, in full safe condition, and without a second person to assist*".

The decomposition of *Func(f)*, as resulting from functional analysis, will identify the following sub functions:

f_1 = convert electricity to air flow,

f_2 = convert electricity to heat,

f_3 = transfer heat to air flow,

f_4 = start/stop the air flow,

f_5 = supply air flow,

f_6 = protect the user and

f_7 = handling the dryer.

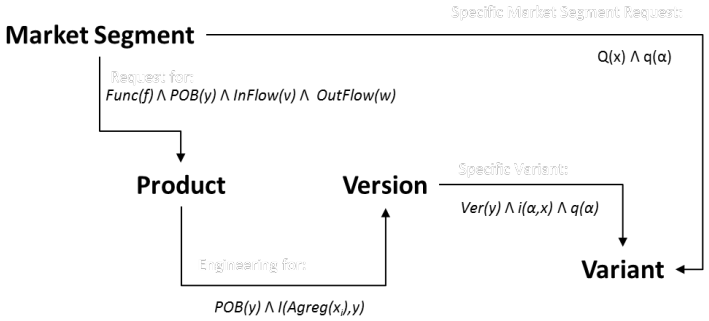

Figure 3. Product, Version and Variant dependencies with Market Segment

Considering eq. (7), the product function for the case of hair dryer example becomes:

$$Func(f) = Agreg(f_1, f_2, f_3, f_4, f_5, f_6, f_7) \qquad (11)$$

The equation above represents the decomposition describing the functional analysis. The decomposition above serves to define the product structure: sub function f_1 will request a fan to be fitted in the product structure, sub function f_2 will request for a heater, aggregation of f_1, f_2 and f_6 will request that electricity to be supplied by a power standardized plug having a coherent interface with available wall-socket and so on. The engineering design had solved these sub functions, the Equations 1, 2 and 4 were met, and the model 1920 shown in Figure 4 depicts the product on the market as first released. From the assigned qualities point of view, in the very beginning models it is expected that assigned qualities do not demonstrate advanced improvement or refining. It can be observed that the model 1920 is fully functional, but aesthetics and ergonomics are not implemented.

The model 1950 offers a changed sub function f_4 = selecting air flow power (against start/stop hair dryer power), and add the sub function f_8 = aesthetic. By refining the product design, the new model offers a better surface condition, a better aesthetic and a new material for the sub function f_7 along with the possibility of selecting the desired power. Nevertheless, the aggregation of these sub functions will keep unchanged the function *Func(f)*, and based on eq. (9), the new model will be defined as a new version of the "hair dryer" product. The model 2000 will be defined also as a new version of the same product. Usually, the versions exist one at a time, so model 2000 will replace model 1950 and so on. Another sub function added is the marketability. It is should be noted that the power of the hair dryer continuously increased from 100 W to 2000 W, mainly as an expression of marketability sub function. The variants exist for every version defined in Figure 4 as, for example, the power supply of 110 V or 220 V, line frequency of 50 or 60 Hz as qualia of respective product qualities. Variants exist simultaneously, but these are related to the product version. In terms of product design researchers, the product version is defined as the evolution of a certain product in the sense of improving and replacing the existing one.

Essentially the product version represents a new revision of that product (Westfechtel, 2001), (Kropsu, 2011); whilst the product variant represents a sub version of a certain product which satisfies a specific market, business or engineering need, essentially the variant extends a product family. Once a product records both variants, the product is called configurable product. For a given product and a given version, more variants exist in parallel (Zha, 2006), (Westfechtel, 2001). Based on these definitions, it can be said that a product version can differ from a previous one by the fact that it brings added value to the product by making a modification in shape, dimension, material, number of components, etc.

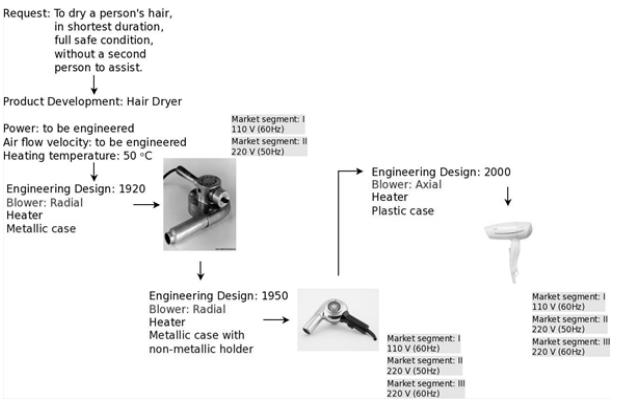

Figure 4. Hair dryer – the product development
Source: pictures from https://www.pinterest.com

6 CONCLUDING REMARKS

Several ontological definitions for product, versions and variants are proposed on this paper. These criteria are necessary for a better understanding of the endurants, perdurants, qualities and qualia entities.

The definition of the product, based on DOLCE ontology and STEP master identification, includes both perspectives of the user and the manufacturer. Also, the dependencies across entities forming the above ontology are important in order to understand the way in which an endurant participates in a perdurant. The research is focusing on the meaning of the specific entities. The models proposed for product, versions and for variants should better represent the real world.

7 ACKNOWLEDGEMENTS

This work is supported by the Sectorial Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and the Romanian Government under the contract number POSDRU/159/1.5/S/137390.

8 REFERENCES

►Borgo S., Masolo C. (2009). *Foundational choices in DOLCE*, Handbook on Ontologies, International Handbook on Information Systems, 2nd edition, Staab S., Studer R. eds., Springer Heildelberg Germany.

►Borgo S., Carrara M., Garbacz P., Vermaas P. (2009). *Towards the Ontological Representation of Functional Basis in DOLCE*, available at: http://www.loa.istc.cnr.it/old/Papers/2_garbacz.pdf Accessed 2014-09-25.

►Borgo S., Carrara M., Garbacz P., Vermaas P. (2010). Formalizations of Functions within the DOLCE Ontology, Tools and Methods of Competitive Engineering,Vol.1, pp. 113 -126.

►Gangemi A., Guarino N., Masolo C., Oltromari A., Schneider L. (2002). *Sweetening Ontologies with DOLCE, Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*. Lecture Note in Computer Science, Vol 2473, pp 166 – 181.

►*** (2005). ISO 10007:2005 – Sisteme de management al calităţii, Linii directoare pentru managementul configuraţiei . ASRO.

►*** (2008). ISO 9001:2008 Quality management systems – Requirements. ASRO.

►Kitsios F. (2000). *Product Design and Development*. Report produced for the EC funded project, Technical University of Crete.

►Mascardi V., Cordì V., Rosso P. (2007). A comparison of Upper Ontologies, Proc. Conf. on Agenti e industria: Applicazioni tecnologiche degli agenti software, WOA07, Genova, Italy.

►Masolo C., Borgo S., Gangemi A., Guarino N., Oltromari A., Scgneider L. (2008). *WonderWeb Deliverable D17, technical report.* doi=10.1.1.11.4243.

►Maurin Anna-Sofia (2014). *Tropes*. The Stanford Encyclopedia of Philosophy (Fall 2014 Edition), Edward N. Zalta (ed.), available at: http://plato.stanford.edu/archives/fall2014/entries/tropes/ Accessed 2015-01-20.

►Mascardi V., Cordì V., Rosso P. (2007). A comparison of Upper Ontologies, Proc. Conf. on Agenti e industria: Applicazioni tecnologiche degli agenti software, WOA07, Genova, Italy.

►Kropsu-Vehkapera H., Haapasalo H., Jaaskelainen O., Phusavat K. (2011). *Product Configuration Management in ICT Companies: The Practitioners' Perspective.* Technology and Investment, Vol. 2, pp. 273-285.

►Ungerer M., Buchanan K. (2002). *Usage Guide for the STEP PDM Schema V1.2 Release 4.3*. PDM Implementor Forum.

►Westfechtel B., Munch B., Conradi R. (2001). *A Layered Architecture for Uniform Version Management*, IEEE Transactions on Software Engineering, Vol. 27, No. 12, pp. 1111-1133.

►Zha X., Sriram R. (2006). Platform-Based Product Design and Development: A Knowledge Intensive Support Approach. Knowledge-Based Systems, Vol. 19, Iss. 7, pp.524-543.