Databases and Information Systems X
A. Lupeikiene et al. (Eds.)

doi:10.3233/978-1-61499-941-6-143

A Semantic Model for Product Configuration in Timber Industry

Hele-Mai HAAV[1] and Riina MAIGRE
Department of Software Science, Tallinn University of Technology, Akadeemia tee 15a, 12618 Tallinn, Estonia

Abstract. Nowadays, customers require customized products instead of standard offers and in many cases prefer web shops and market places for purchasing goods and services. Therefore, intelligent product configuration tools are great value for many companies. One of the main components of these tools is a configuration model. In order to integrate the configuration model to the semantic web new approaches to the representation of product configuration knowledge are needed. In this paper, we present a layered ontology modelling framework for building semantic product configuration models and provide the corresponding methodological approach to product configuration. The approach enables building distributed product configurators that use semantic configuration models in the form of web ontologies and Shapes Constraint Language (SHACL) to validate integrity constraints of individual configurations. SHACL is also used to query the model and to represent additional constraints required by different applications. The provided method is demonstrated on the basis of the real case study from timber industry that specifically covers thermally modified timber (sawn wood) products.

Keywords. Ontology, OWL, product configuration, SHACL, SPARQL

1. Introduction

Web marketing and sales of products in B2B or B2C markets are gaining more and more importance for any industry. Today, many customers require customized products instead of standard offers. This makes user oriented product configuration tools great value for improving manufacturing, marketing and sales processes as well as making possible to grow sales amounts.

Product configuration software enables a customer to choose parameter value options for building a customized product that matches their desired configuration (see e.g. iMark[2], Studio_shed[3], and eJeeva[4]). For example, in automotive and fashion industries such product configurators are well used like in Tesla[5] and Nike[6]. In contrast, in timber industry and particularly for sawmill products this is not the case. In this

---

[1] Corresponding Author, Hele-Mai Haav, Department of Software Science, Tallinn University of Technology, Akadeemia tee 15a, 12618 Tallinn, Estonia, E-mail: helemai@cs.ioc.ee.

[2] www.iMark.com

[3] https://www.studio-shed.com/configurator

[4] http://ejeeva.com/product-configurator

[5] https://www.tesla.com/models/design

[6] https://www.nike.com/us/en_us/c/nikeid

paper we try to bridge this cap by providing a case study from the domain of timber industry covering thermally modified timber (sawn wood) products. The case is related to our industrial partner Thermory AS that likes to solve some of its B2B and B2C problems using an intelligent product configurator. In this domain, the product features (parameters) as well as constraints that the customized (ordered) product should satisfy are more complex than in the case of ordinary sawmill products.

Product configuration software tools need to work on the basis of a knowledge base (i.e. configuration model) including product configuration knowledge provided by domain experts. In general, product configuration knowledge represents product structure and restrictions. Traditionally, the required knowledge is presented in the form of rules or constraints. Therefore, product configuration engines are rule engines or constraint solvers accordingly [1]. Many of them are connected or can be integrated to traditional Enterprise Resource Planning (ERP) systems.

Recently, there have been proposals to make use of semantic web technologies in the field of building product configuration software [2, 3, 4]. Product configuration knowledge can be represented in the form of a formal ontology expressed in Web Ontology Language (OWL) [5]. Semantic Web Rule Language (SWRL) [6] can extend the OWL for providing capabilities to express constraints that are not expressible as OWL axioms. In [3] all product configuration constraints are expressed as SWRL rules making the solution similar to rule based approaches.

Main benefits of using ontology-based configuration models are the ability to express complex constraints and cardinalities of properties in OWL that is based on rigorous semantics. This makes it possible to reason on the ontology using inference engines and consequently ontologies can be interpreted by machines.

Nevertheless, ontology based configuration models in proposals mentioned above are rather standalone ontologies and are not well integrated to current semantic web development. Another issue is that there are not enough reusable product ontologies or semantic web vocabularies available on the web. For example, there is not any product and product configuration ontology available that can be reused for timber industry.

Therefore, the goal of this paper is to provide a semantic product configuration model and the corresponding methodological approach to product configuration as well as to demonstrate it in the domain of timber industry. The configuration model is intended to be well integrated to the existing semantic web vocabularies and used as a knowledge base for the interactive web-based configuration of timber products but also for other applications like semantic HTML mark-up, web marketing, sales, etc.

In our previous work [7], we have proposed standalone ontology for capturing variance knowledge of thermally modified wood products. In this paper, we extend this work by providing a general modelling framework for building and usage of semantic product configuration models for timber industry.

The proposed semantic model has hierarchical structure consisting of meta-model, model and instances levels. Schema.org vocabulary[7] is considered on meta-model level, specific timber industry product ontology is located on model level being a specification of meta-model. Constraints are expressed on model level as class expressions. Finally, instances level represents particular products with their parameters.

[7] https://schema.org

The semantic configuration model is encoded in RDF[8] and OWL. The Description Logic (DL) reasoner Pellet [8] is used for ontology reasoning but other DL reasoners can be used as well.

The semantic configuration model is accessible via SPARQL [9] and/or Shapes Constraint Language (SHACL) that is W3C recommendation since 2017[9]. Some configuration constraints that involve computations as well as application specific constraints are expressed in SHACL.

There are several benefits of such a model. First of all, it is reusable in many applications as well as by humans and can be used as a reference model for thermally modified wood products domain.

Second, using Schema.org on meta-modelling level enables integration to the semantic web vocabulary used by many search engines (e.g. Google) and e-commerce sites (e.g. eBay).

The paper is structured as follows. In Section 2 we analyse related works. Section 3 is devoted to our original semantic product configuration model. Section 4 provides the methodological approach of our product configuration framework. In Section 5 we demonstrate our framework on the case study of timber industry. Conclusion and future work are presented in Section 6.

2. Related Works

One of the first works (published in late 90s) that uses DL based knowledge bases for product configuration is [10]. They have built configurators based on DL based knowledge representation system CLASSIC [11] for a number of large telecommunications products sold by AT&T and Lucent Technologies.

At the same time the work towards a general ontology of configuration was developed in order to reuse and share configuration knowledge [12]. This ontology includes concepts like components, attributes, resources, ports, contexts, functions, constraints, and relations between these. It is formalised in Ontolingua [13] based on KIF [14] that lacks reasoning mechanism for checking the consistency of a knowledge base that is available in DL based languages.

In [4] an ontology-based product configuration model was developed and formalized using OWL and SWRL. A similar approach can be found in [3], where focus is on the semantics of constraints of product configuration that cannot be expressed by OWL. They provide a rule based ontological formalism for describing product structure and constraints of a product configuration and checking its validity.

Interesting relationships can be found between feature-oriented domain analysis (FODA) [15] used basically for software line production (SLP) and ontology based product configuration models. The authors of [16] analysed similarities and differences of feature models of FODA [15] and ontology based domain analysis methods. According to their work, similarities include using a concept vocabulary, enabling the expression of property and class hierarchies, and providing a constraint definition capability. In FODA, the latter is used for variability reduction but in ontology based domain analysis constraints are used for the description of property restrictions in class expressions. Both analysis methods allow to describe semantics of a domain and can be

[8] https://www.w3.org/RDF/
[9] https://www.w3.org/TR/shacl/

represented in machine readable form. Therefore, in [15] it is decided that ontologies could effectively replace FODA models. As the advantages, ontology based analysis provides more expressive language than FODA and includes additional capabilities like reasoning and querying. We may conclude that from the conceptual point of view, there are similarities between variability management in product configuration and SLP.

In addition, there is an interesting attempt to express product configuration data as linked data published on the web [17]. This proposal is related to the Renault AS configuration services. Their configuration model is represented in the form of linked data (i.e. RDF) as configuration ontology based on GoodRelations vocabulary [18].

Using Schema.org vocabulary on meta-level is related to works on extension of Schema.org vocabulary with different vocabularies for various specific product types like, for example, in fashion industry [19]. Our work differs from their approach in that we use the Schema.org Product type only on meta-level of the configuration model. They do not deal with configuration problems.

3. A Semantic Product Configuration Model

3.1. Options for Semantic Modelling of Products

Semantic modelling of product data and knowledge is an active area in semantic web technology development in order to make it possible that computers can process product data published on the web. This is to increase the visibility of products and services in the modern search engines and other novel web applications.

For that purpose, W3C issued a document about product modelling using semantic web technologies[10] in 2009. This document provides principles for product modelling in the form of reusable, generic upper product ontology. The ontology covers main concepts for modelling quantities, units and scales as well as product structure. The document also describes general modelling principles for semantic modelling of product knowledge. It recommends importing the upper product ontology and specialising its concepts for any specific product ontology.

Currently, there are several standardized vocabularies available that also include product related concepts. One well-known vocabulary is e-business oriented GoodRelations that provides concepts and relationships for product, its price, and company data [18]. GoodRelations ontology can be combined with eClassOWL[11] that is OWL ontology for describing the types and properties of products and services described in industry standard eCl@ss[12] to be used for e-business.

An alternative vocabulary that is developed for adding semantic mark-up to the HTML representation of web pages and used by the major search engines like Google, Microsoft, Yandex and Yahoo! is Schema.org (launched in 2011).

Schema.org is semantically lightweight vocabulary having rather broad scope of widely used types including also the Product type[13]. The goal of Schema.org is to provide web content creators possibility to enrich their HTML code with Schema.org metadata, making human readable information also machine readable. Schema.org

[10] https://www.w3.org/2005/Incubator/w3pm/XGR-w3pm-20091008
[11] http://www.heppnetz.de/projects/eclassowl/
[12] https://www.eclass.eu/en.html
[13] https://schema.org/Product

vocabulary is officially represented in RDFa[14] that is a W3C standard which enables embedding RDF in HTML. Schema.org types can be extended according to proposals from the community to the Schema.org committee.

Last but not least, there is the Productontology.org[15] that extends Schema.org and GoodRelations vocabularies with links to hundreds of thousands precise definitions for types of product in Wikipedia.

3.2. A Proposed Ontology Modelling Framework

In this paper we present layered ontology modelling framework for building semantic product configuration models for products that do not have components (or assemblies) but rather many variant parameters that characterize the products and can be specified by a customer. This makes the model simpler than general product configuration model provided in [12]. We demonstrate the framework in the domain of timber industry.

Proposed modelling framework has 3 layers depicted in Figure 1 as follows: meta-model, model and instances levels.

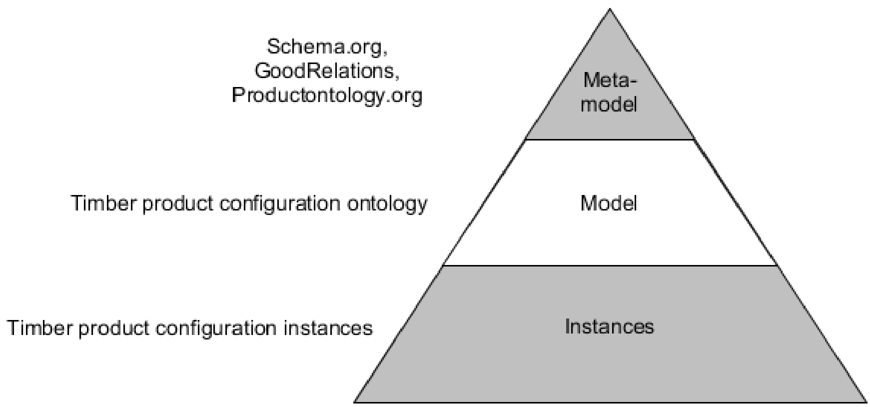

Figure 1. The layered semantic configuration model for the timber industry case study.

3.2.1. Meta-model Level

Ontology that is chosen to be used on meta-model level should describe general concepts and relationships of the product configuration domain and thus not to be specific to any product or product configuration method (system). Rather to develop meta-ontology from scratch, we looked for existing suitable ontologies in the product configuration domain. There are some candidate ontologies reported in literature like general ontology of configuration in [12] and its extension in [4]. These ontologies include main classes as follows: part, assembly, port, function, constraint, attribute, etc. Difference of these two approaches is in formal representation language used to describe ontology. The first is represented in Ontolingua [13] that does not have tractable method for checking consistency of knowledge base and the second in OWL

[14] https://schema_org_rdfa.html

[15] https://Productontology.org

that is based on DL and has it. However, we did not choose the ontology in [4] for meta-model level.

There are several reasons for that as follows:

- We do not need such a complex ontology as we do not intend to model parts and assemblies, etc.
- This ontology is not published on the web what means that it cannot be widely used for reusing and sharing knowledge by applications.

We decided to use Schema.org on meta-level instead of a specific product configuration domain ontology as Schema.org includes the concept of product and its common properties. In principle, alternative options can be also used as listed in Section 3.1. Our main justification of the choice is that we like to benefit from search engines work for marketing and sales purposes.

We distinguish namespaces of meta-model and model levels in general. On meta-model level we use the following prefix definition: schema: http://schema.org/. On model level we use namespace prefix timber (for the timber industry case study) and the following prefix definition: timber: http://www.ioc.ee/ontologies/timber.

We use generic concept schema:Product on meta-model level as a superclass of the timber product class allowing to reuse concepts and relationships defined for it in Schema.org by concepts specified on model level using timber product classes.

Table 1 lists the properties of the schema:Product type that describe basic common information about any product and we recommend to reuse them on meta-level to specify also timber products (see for more properties in schema.org/Product).

Table 1. A list of properties of the Schema.org Product type applicable to a timber product description.

Property name	Schema.org name
Product name	schema:name
Product description	schema:description
Product image	schema:image
Product link	schema:url
Product manufacturer	schema:manufacturer
Product price	schema:price
Product identifier	schema:identifier

Generic Product type from Schema.org does not provide concepts and relationships specific for timber industry as well as those needed for building timber product configuration models. Therefore, we need to provide specific knowledge on model level of our framework.

3.2.2. Model Level

A product specific ontology is considered on model level. A specific product is represented as a subclass of schema:Product. It inherits properties from the schema:Product type.

The proposed upper level class hierarchy for timber products is represented in Figure 2 in Turtle syntax[16].

[16] https://www.w3.org/TR/turtle/

```
Declaration of namespaces

@prefix timber: < http://www.ioc.ee/ontologies/timber/> .
@prefix schema: <http.//schema.org/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .


Declaration of classes

timber:Timber   rdf:type owl:Class .
timber:ModifiedTimber   rdf:type owl:Class .
timber:ThermallyModifiedTimber   rdf:type owl:Class .
timber:ChemicallyModifiedTimber   rdf:type owl:Class .
timber:RoughTimber   rdf:type owl:Class .


Declaration of subclasses

timber:Timber   rdfs:subClassOf    schema:Product .
timber:ModifiedTimber   rdfs:subClassOf    timber:Timber  .
timber:ThermallyModifiedTimber   rdfs:subClassOf    timber:ModifiedTimber  .
timber:ChemicallyModifiedTimber   rdfs:subClassOf    timber:ModifiedTimber  .
```

Figure 2. Upper level class hierarchy for the timber industry case study.

From the timber manufacturing point of view, a configurable product belongs to a product family that includes a large number of possible products characterized with certain properties. The configuration process should ensure that the individual product that is to be produced does not violate these properties. In this case, it is not feasible to describe all the individual products in the configuration model. Instead, the product family can be described as a class including family specific properties and constraints and a description of each product configuration is derived by configurator on the basis of the model and a user input. Therefore, in addition to classes of the upper ontology of timber products, a timber product and its family specific properties and constraints that are needed to be expressed in configuration model are considered on model level.

Products of timber industry are characterized with high number of product variants. Schema.org has property schema:isVariantOf through its subtype ProductModel to model variants of product models. However, it requires materializing all available configurations explicitly as variants of a given base model. The same approach is used by traditional ERP systems (databases). Schema.org does not include a generic mechanism for modelling product variants (or configuration options) and does not support rule based representation and generation of variants. All variants should be materialized using for example SPARQL CONSTRUCT [9] queries. This is what we like to avoid as unfeasible.

In contrast to the above mentioned approach, we provide in this paper the timber product configuration ontology that includes products taxonomy and axioms to represent valid timber product families and variants. This supports modelling of configurations of various timber products without prior materialization of all variants.

Terminological knowledge about product families and product variants is expressed as the product configuration ontology class hierarchy, object property and data property definitions and class expressions (TBox in DL). For each product family a subclass of the class Timber is defined as a complex class in OWL. Knowledge about product variants is defined in subclasses of the corresponding product family class (for example, see the ThermallyModifiedTimber class and its subclasses in Figure 4). All subclasses of classes are disjoint. Object and data properties express either standard or variant parameters of a product. They are associated with the class Timber having it as a domain.

Constraints are represented as property restrictions in complex class expressions in OWL. Class expressions define a set of individuals belonging to the class.

3.2.3. Instances Level

On instances level of configuration model, individuals (ABox in DL) are used to represent product features with predefined value instances like value choices of variant parameters (e.g. profile D4). Other product parameter values are represented as data property values.

In addition, user input requirements are considered as distinct individuals of a class of a certain product with provided values for variant parameters. They are used for validation of the configuration result.

3.3. Validation of Model

The validation of the configuration model means checking whether a product configuration model is consistent or not. This procedure uses standard ontology consistency check. DL reasoners use Open World Assumption (OWA) for reasoning meaning that the model may be incomplete and new knowledge may be added that necessarily is not false. This is good for checking partially defined product variants.

During the evolution of ontology (according to the evolution of product variants) its consistency needs to be checked again by a DL reasoner before it can be used for the validation of parameters of an individual ordered product.

4. A Proposed Methodological Approach to Product Configuration

The approach proposed in this paper is based on a traditional scheme of configuration framework where given a configuration model and a user input (requirements) the configuration software derives a valid configuration or a valid optimal configuration. We enrich this basic scheme with semantic web technologies enabling to build distributed product configurators that use semantic configuration models in the form of web ontologies and SHACL to query the model and validate the configuration results. A general scheme of our methodological approach is represented in Figure 3.

The configuration model captures knowledge of variants of products in the form of the semantic product configuration model as described in Section 4. In the case of timber industry, the configuration model is designed so that a valid configuration inherits a number of parameters and their values from product family classes or product variant classes (for example, see Figure 4). The configuration model is represented in OWL and can be accessed by applications (including configuration tools) by any OWL API or using SPARQL and SHACL or Apache Jena[17].

[17] https://jena.apache.org/

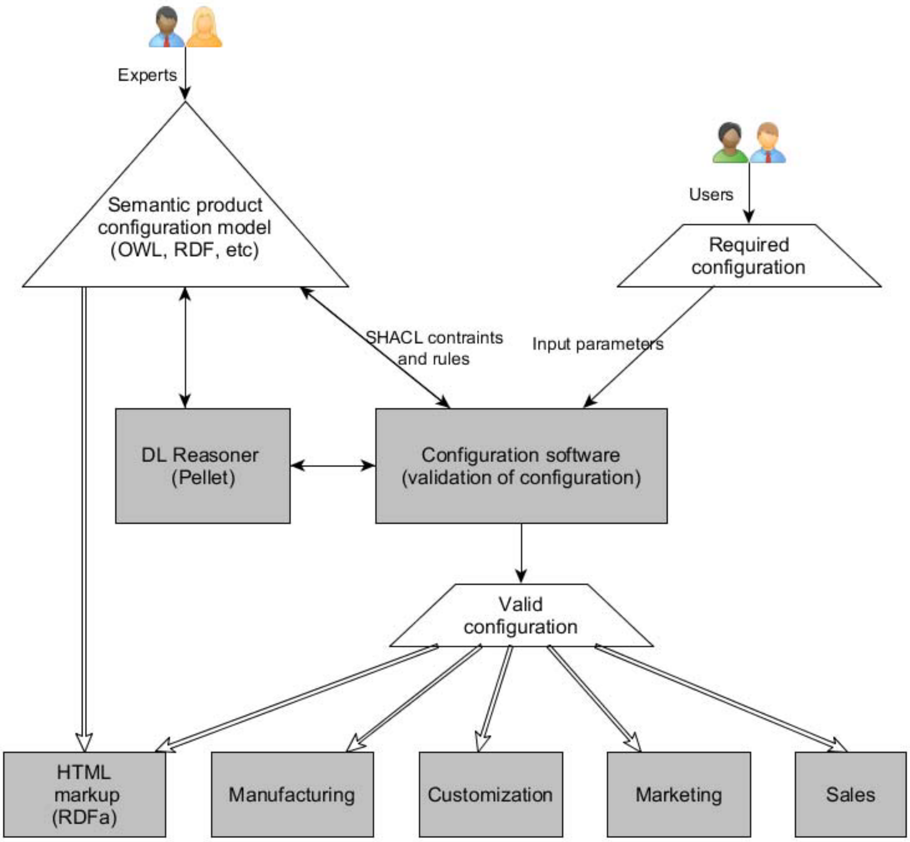

Figure 3. General scheme of the proposed configuration method.

Validation of the configuration model is performed during its design time. Customer needs are represented in conformance with product class instance(s) of instance level of the modelling hierarchy (see Figure 1). In order to output a valid configuration the configuration software needs to check that the configuration is consistent with respect to the configuration model and that its description is complete according to the configuration model.

DL reasoning is used to check whether the user input requirements (variant parameter values) define an individual product that satisfies a product configuration model. Reasoning is also used for inferring standard parameters of a user input product according to the given product hierarchy.

However, the verification whether the user input is complete according to the configuration model cannot be done only by using DL reasoners as this requires Closed World Assumption (CWA) reasoning.

According to our methodological approach consistency check is performed by a DL reasoner like Pellet and completeness check is done using SHACL. The latter is also applied for checking additional constraints and performing computations with parameters of the valid product configuration if necessary.

If both checks give positive results then configuration software gives a description of a valid configuration as output.

Valid product configurations can be used by many different applications (as well as product configuration model itself is reusable). For example, the semantic mark-up of HTML pages, manufacturing to order B2B and B2C applications, custom product builders integrated to online shops and web markets are possible application options.

Our semantic product configuration model can be extended using content from different databases of product catalogues in order to support semantic interoperability like in [20].

5. The Case of Timber Industry

5.1. Scope and Purpose of the Case Study

We illustrate proposed semantic product configuration model and method on the basis of the real case study related to the thermally modified wood producer Thermory AS[18]. It is an Estonian company specialising in thermally modified solid wood flooring, decking, and cladding. Thermory manufactures about 400 different products, which are ordered in large number of variants that makes the expression of the product variant knowledge and validation of the product configuration very important. Thermory AS uses chemical-free thermal modification process, where properties of wood are altered using only heat and steam[19].

Depending on the customer order requirements, products can have different thermal modification levels (peak temperatures 190° and 215°) that depend mainly on wood species. Boards have additional variations in dimensions, length, profile and suitable clips. Available profiles and suitable clips depend on dimensions of the board.

In the case study we consider a set of thermally modified wood product families like ash decking and cladding boards, pine decking and cladding boards and spruce decking boards that form the largest share of the production of Thermory.

5.2. The Product Configuration Ontology

For creation of model level of the timber product configuration model, we first classify products to product families that are subclasses of the ThermallyModifiedTimber class from the upper timber products ontology shown in Figure 2. Each product family includes a number of different products (i.e. parent products) that can have variants according to the values of some parameters. This is modelled using product variant classes as subclasses of a product family class.

Product family class sets up values and constraints for standard parameters inherited by corresponding product variant classes that represent specific variations of a product belonging to a product family class taking into account the values of variant parameters (see Figure 4).

Variant parameters which have numerical or textual values are modelled using corresponding data properties as follows: hasUsage, hasThermalModification, hasWoodSpecies, hasThickness, hasWidth, hasAreaMM2, hasMinimalLength,

[18] http://thermory.com/en/kontakt/about-company
[19] https://www.thermoryusa.com/modification

hasMaximalLength, hasActualLength, etc. For example, an actual length of an ordered product is a variant parameter that has associated constraints expressing that its value should be between minimum and maximum length given as product parameters. For some products, the actual length parameter can obtain a value only from a specified set of valid values.

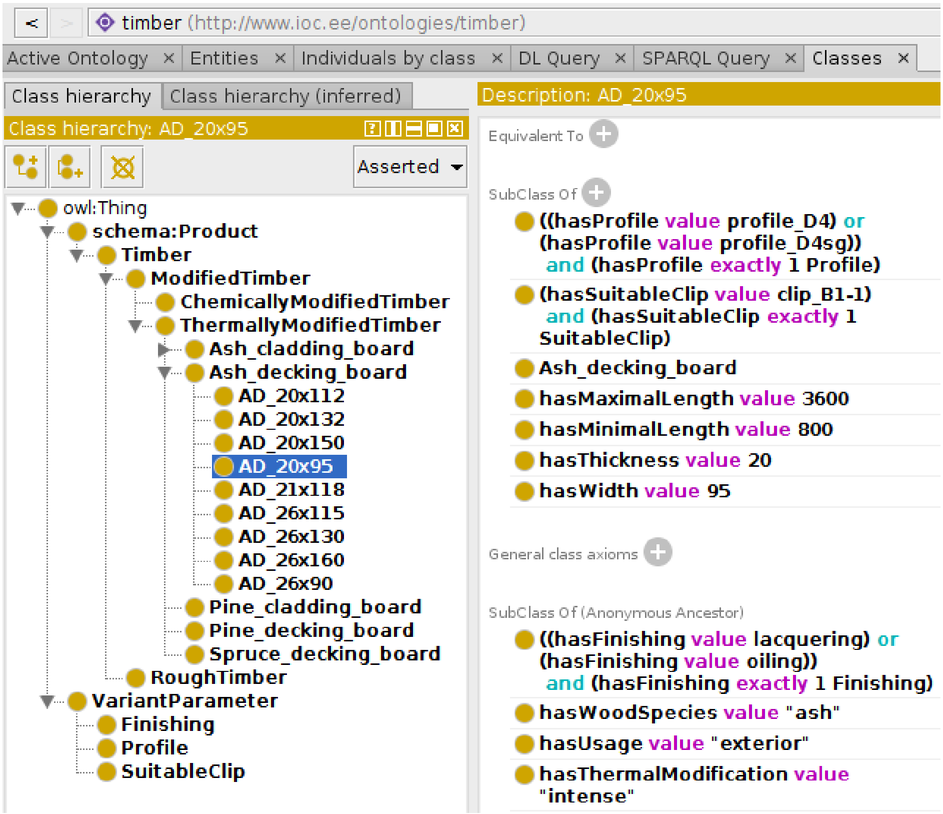

Figure 4. An excerpt of the product ontology class hierarchy and examples of class expressions.

For other parameters there is the class VariantParameter in the ontology. It has currently three subclasses as follows: Finishing, Profile, SuitableClip. Individuals of these classes are created to the instances level of model for modelling specific parameter choices. They are linked to the individuals of product classes via the corresponding object properties as follows: hasFinishing, hasProfile and hasSuitableClip.

The product configuration ontology class hierarchy corresponding to the case study is presented in Figure 4. The class hierarchy contains disjoint classes for product families, product variants and variant parameters. A product family class is defined as a complex class with class expression including data property value restrictions for standard parameters of the product family products.

In addition, according to the specific product family, this class expression may define common property restrictions over object properties corresponding to variant parameters of a product family. Product variants are defined as subclasses of a product family class and their class descriptions specify only data property values and object property restrictions that correspond to the specific variant parameters of this product. They inherit common properties from their product family class.

For example, in Figure 4, the Ash_decking_board product family class that is the superclass of the product variant class AD_20x95 includes property restrictions over hasFinishing object property and value restrictions on the hasUsage, hasThermalModification, and hasWoodSpecies data properties. In Figure 4, we use the format of the ontology editor Protégé[20] to illustrate property restrictions as it is easy to read and short.

Ash_decking_board defines the class of products as the set of individuals that are linked to a finishing option by the hasFinishing property by using the cardinality restriction, which specifies that exactly one element can be in this relation. In addition, this class expression says that the class contains individuals that are connected by the hasFinishing property with an individual lacquering or oiling. In the similar way the specific class expressions are defined for the product variant class AD_20x95 for the hasProfile and the hasSuitableClip object properties.

Using such principles of construction of class expressions makes it possible to use a DL reasoner to automatically infer predefined data property values for a user input product requirements as well as to check if this product is consistent with ontology (i.e. does it satisfy the conditions given in the class expressions).

5.3. Validation of the Configuration Result

One of the main tasks of configuration software is to guarantee that the resulting configuration is valid. A configuration is valid if it is consistent with respect to configuration model and complete. A configuration is consistent if it does not violate anything in the model (i.e. product configuration ontology). We check it with the DL reasoner Pellet (see also Section 4).

It is not easy to perform OWL data validation in the case of checking completeness of a configuration (individual in Abox) that assumes complete knowledge. For that purpose we need to combine OWA reasoning and CWA constraint checking which assumes that the specification information is complete. In general, this is not well supported by now. Therefore, we express and validate integrity constraints in SHACL and integrate completeness check in this way to our approach.

According to our methodological approach, the validation of a configuration consists of two steps as follows:

- Checking that an individual configuration is consistent with respect to model.
- Checking that an individual configuration is complete according to given model.

These two steps can be repeated as necessary, until the configuration is both complete and consistent.

In order to check completeness of the user defined product configuration, we first use Pellet for OWL inference, which adds properties inherited from product variant classes to the configuration. This inferred configuration is checked for completeness

[20] https://protege.stanford.edu/

using TopBraid SHACL API[21] that supports both SHACL Core and SHACL-SPARQL validation, as well as SHACL Rules. In order to check that new product configurations are complete we defined a SHACL shape ProductCompletenessShape, where we describe property constraints for user inserted object and datatype properties that a valid product configuration has to have, such as timber:hasProfile, timber:hasSuitableClip, timber:hasFinishing, timber:hasOrderedQuantityPcs and timber:hasActualLength. These constraints allow us to check both cardinality and datatype for user defined properties. In addition we created SHACL SPARQL constraints to check that the user inserted value of the timber:hasActualLength property is between timber:hasMaximalLength constraint and timber:hasMinimalLength constraint, that were inherited from corresponding product variant classes in the previous step. We do not define constraints for inherited property values in ProductCompletenessShape, since their consistency and completeness are guaranteed with correctness of ontology.

```
timber:ProductCompletenessShape
        rdf:type sh:NodeShape ;
        sh:targetClass schema:Product ;
        sh:property [
                sh:path timber:hasProfile ;
                sh:maxCount 1 ;
                sh:minCount 1 ;
                sh:class timber:Profile ;
        ];
        ...
        sh:property [
                sh:path timber:hasOrderedQuantityPcs|;
                sh:maxCount 1 ;
                sh:minCount 1 ;
                sh:datatype xsd:integer ;
        ];
        sh:property [
                sh:path timber:hasActualLength ;
                sh:maxCount 1 ;
                sh:minCount 1 ;
                sh:datatype xsd:integer ;
        ];
        sh:sparql [
                rdf:type sh:SPARQLConstraint ;
                sh:message "Max length constraint violated" ;
                sh:prefixes timber: ;
                sh:select """
                SELECT $this
                WHERE {
                        $this timber:hasActualLength ?length .
                        $this timber:hasMaximalLength ?maxlength .
                        FILTER ( ?length > ?maxlength )
                }
        """
        ];
        sh:sparql [
                rdf:type sh:SPARQLConstraint ;
                sh:message "Min length constraint violated" ;
                sh:prefixes timber: ;
                sh:select """
                SELECT $this
                WHERE {
                        $this timber:hasActualLength ?length .
                        $this timber:hasMinimalLength ?minlength .
                        FILTER ( ?length < ?minlength )
                }
        """
        ];
.
```

Figure 5. An excerpt from the SHACL shape definition.

[21] https://github.com/TopQuadrant/shacl

An excerpt of the created SHACL shape definition is shown in Figure 5. In this figure SHACL namespace is declared with prefix sh. Other namespaces are declared as shown in Figure 2. The figure illustrates property constraint definition for timber:hasProfile that has to be defined exactly once and its value must be an instance of the class timber:Profile. Consistency check for timber:hasProfile value is done using Pellet in the consistency checking step. Constraint definitions for timber:hasSuitableClip and timber:hasFinishing are identical and are therefore not shown in the figure. For the user defined value timber:hasOrderedQuantityPcs we only check that it is defined exactly once and that its datatype is xsd:integer. Another user defined value timber:hasActualLength has three constraints. First, there is a property constraint that checks if the value of type xsd:integer was defined exactly once. Second, there is a SHACL SPARQL constraint to check that the user defined value is less than timber:hasMaximalLength. A third constraint that we have defined checks that timber:hasActualLength is greater than timber:hasMinimalLength. Note that, timber:hasMaximalLength and timber:hasMinimalLength used in SPARQL constraints have to be inferred in OWL inference step, before SHACL validation can be done.

5.4. Configuration Rules

Not all configuration constraints can be expressed by OWL class expressions and SHACL constraints but in many cases timber calculations are needed. Calculations can be expressed using SHACL rules. Figure 6 shows a definition of SHACL ProductRulesShape with two SHACL SPARQL rule definitions that use SPARQL CONSTRUCT queries in order to calculate value for timber:hasAreaMM2 and

```
timber:ProductRulesShape
        rdf:type sh:NodeShape ;
        sh:targetClass schema:Product ;
        sh:rule [
               rdf:type sh:SPARQLRule ;
               sh:prefixes timber: ;
               sh:construct """
               CONSTRUCT {
                        $this timber:hasAreaMM2 ?areaMM2 .
               }
               WHERE {
                        $this timber:hasActualLength ?length .
                        $this timber:hasWidth ?width .
                        BIND (?length * ?width AS ?areaMM2) .
               }
               """ ;
        sh:condition timber:ProductCompletenessShape ;
        ] ;
        sh:rule [
               rdf:type sh:SPARQLRule ;
               sh:prefixes timber: ;
               sh:construct """
               CONSTRUCT {
                        $this timber:hasOrderedTotalMeters ?totalMeters .
               }
               WHERE {
                        $this timber:hasOrderedQuantityPcs ?pcs .
                        $this timber:hasActualLength ?length .
                        BIND ((?length * ?pcs ) / 1000 AS ?totalMeters) .
               }

               """ ;
        sh:condition timber:ProductCompletenessShape ;
        ] ;
.
```

Figure 6. SHACL rule definitions.

timber:hasOrderedTotalMeters properties. Rules are only applied to the product configuration by SHACL if the configuration conforms to the ProductCompletenessShape definition shown in Figure 5. This ensures that user inserted values for timber:hasActualLength and timber:hasOrderedQuantityPcs are defined according to the constraint definitions. Values, such as timber:hasWidth have to be inferred before the rules can be applied, since we do not define SHACL shape constraints for inferred values.

6. Conclusion

In this paper we proposed a methodological approach to product configuration that is based on the original layered semantic product configuration model.

The configuration model consists of meta-model, model and instances levels. Meta-model level makes it possible to link specific product ontology concepts from model level to the Schema.org Product type or to other shared product vocabularies from the web. Configuration constraints are expressed on model level as class expressions and instances level represents user requirements on product parameters. The configuration model supports modelling of configurations of various products without prior materialisation of all variants. The model is encoded in RDF and OWL.

The methodological approach provided in the paper ensures that the resulting configuration is valid i.e. consistent with respect to configuration model and complete. Consistency check is performed by the DL reasoner Pellet. Completeness check requires CWA check that should be combined with OWA reasoning. We express and validate integrity constraints in SHACL. It is also used for representing configuration constraints that involve computations or that are application specific.

Our methodological approach is demonstrated in the domain of thermally modified wood products produced by Thermory AS. The configuration model can be used for the interactive web-based configuration of timber products but also for other applications like semantic HTML mark-up, manufacturing, web marketing, sales, etc.

Presented semantic configuration modelling framework is general and can be applied in many industries where products do not have complex components and assemblies but rather many variants with different variant parameter values.

Acknowledgements

This work was supported by the Institutional Research Grant IUT33-13 of the Estonian Research Council. We express our gratitude to Rebeka Plees from Thermory for sharing her expert knowledge about variants of thermally modified wood products and Vahur Kotkas for discussions on optimization of timber manufacturing process.

References

[1] J. Tiihonen, M. Heiskala, A. Anderson and T. Soininen, WeCoTin – A practical logic-based sales configurator, *AI Communications* **26** (2013), 99-131.

[2] J. S. Liang, Generation of automotive troubleshooting configuration system using an ontology-based approach, *Computers in Industry* **63** (2012), 405-422.

[3] S. Xuanyuan, Y. Li, L. Patil, Z. Jiang, Configuration Semantics Representation: A rule-based ontology for product configuration. In: *Proceedings of SAI Computing Conference* 2016, pp. 734-741. IEEE (2016).

[4] D. Yang, M. Dong, R. Miao, Development of a product configuration system with an ontology-based approach, *Computer-Aided Design* **40**(8), (2008), 863-878.

[5] B. Motik, et al., OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax., *W3C recommendation*, http://www.w3.org/TR/owl2-syntax, last accessed 2018/03/04.

[6] I. Horrocks, et al, SWRL: A Semantic Web Rule Language Combining OWL and RuleML, *W3C Member submission*, https://www.w3.org/Submission/SWRL, last accessed 2018/03/05.

[7] H-M. Haav and R. Maigre, Domain Ontology for Expressing Knowledge of Variants of Thermally Modified Wood Products, In: *International Baltic Conference on Databases and Information Systems*, pp. 161-171. CCIC Vol. 838, Springer (2018).

[8] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, Y. Katz, Pellet: A Practical OWL-DL Reasoner, *Web Semantics: science, services and agents on the World Wide Web* **5**(2), (2007), 51-53.

[9] S. Harris, et al, SPARQL 1.1 Query Language, *W3C recommendation*, https://www.w3.org/TR/sparql11-overview/, last accessed 2018/03/04.

[10] D. McGuinness and J.R. Wright, An industrial-strength description logic-based configurator platform, *IEEE Intelligent Systems and their Applications* **13**(4), (1998), 69-77.

[11] P. F. Patel-Schneider, D. L. McGuinness, R. J. Brachman, L. A. Resnick, The CLASSIC knowledge representation system: guiding principles and implementation rationale. *ACM SIGART Bulletin* **2**(3), (1991), 108-113.

[12] T. Soininen, J. Tiihonen, T. Mannisto, R. Sulonen, Towards a general ontology of configuration. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing AI EDAM* **12**(4), (1998), 357-72.

[13] T. R. Gruber, Ontolingua: A Mechanism to Support Portable Ontologies, Technical Report. Stanford University (1992).

[14] M. R., Genesereth and R. E. Fikes, Knowledge Interchange Format Reference Manual. Technical Report. Computer Science Department, Stanford University (1992).

[15] M. Acher, P. Collet, P. Lahire, R. France, Comparing approaches to implement feature model composition. In: *Proceedings of 6th European Conference on Modelling Foundations and Applications*, ECMFA 2010, pp 3-19. LNCS Vol. 6138, Springer (2010).

[16] C. Ines, M. Crepinšek, T. Kosar, M. Mernik, Ontology driven development of domain-specific languages, *Computer Science and Information Systems* **8**(2), (2011), 317-342.

[17] E. Chevalier and F-P. Servant, Product customization as linked data, *Extended Semantic Web Conference*, ESWC 2012, pp. 603-617. LNCS Vol. 7295, Springer (2012).

[18] M. Hepp, GoodRelations: an ontology for describing products and services offers on the web, *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management* (EKAW2008), pp. 332-347. LNCS Vol. 5268, Springer (2008).

[19] A. Stolz, M. Hepp, A. Hemminger, Representing fasion product data with Schema.org: approach and use cases. In: *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, pp. 254–272. LNCS 10574, Springer (2017).

[20] C. Ardito, B. R. Barricelli, P. Buono, M. F. Costabile, R. Lanzilotti, A. Piccinno, S. Valtolina, An ontology-based approach to product customization. In: *International Symposium on End User Development*, pp. 92-106. LNCS Vol. 6654, Springer (2011).