INTERNATIONAL DESIGN CONFERENCE – DESIGN 2020
https://doi.org/10.1017/dsd.2020.139

TOWARDS AUTOMATED CLASSIFICATION OF PRODUCT DATA BASED ON MACHINE LEARNING

S. Schleibaum [1,✉], S. Kehl [2], P. Stiefel [2] and J. P. Müller [1]

[1] Technische Universität Clausthal, Germany, [2] Volkswagen AG, Germany

✉ soeren.schleibaum@tu-clausthal.de

Abstract

Modern machine learning methods have the potential to supply industrial product lifecycle management (PLM) with automated classification of product components. However, there is only little work in the literature on this topic. We propose to apply supervised machine learning on component meta-data. By analysing an industrial case study, we identify requirements and opportunities for automating classification, e.g. in part numbers and product structures. We validate our novel approach through a classification experiment comparing four machine learning methods on a realistic component dataset.

Keywords: product lifecycle management (PLM), machine learning, classification, product structure, product data management (PDM)

1. Introduction

Complex products are composed of components, which themselves are collections of domain-specific documents such as geometric data in design (Kehl et al., 2015). During development, a design engineer receiving an order to construct a component for a new product, e.g. a brake pad, first looks for existing brake pads that already meet the requirements, in example specific length and thickness. The same applies to the retrieval of spare parts with similar or identical characteristics. In order to identify suitable brake pads, each existing brake pad has to be retrievable by length and thickness. Such set of properties can be described as one *class* (Domingos, 2012). By use of several classes, all components can be differentiated according to their properties. Due to partitioning, retrieving components can be supported by a given *classification*.

In addition to finding existing components, classification can be helpful to support integrated product development processes (PDPs) during and after the design phase by linking different storage systems and methodologies. Example: To manage product data, engineers use static product structures, such as bills of materials (BOMs) as data backbones (Adolphy et al., 2015). There are various types of BOMs in different phases of the development process, e.g. Engineering BOM and Manufacturing BOM, with different focus and structure (Tekin, 2014).

To support a continuous development process, these different BOMs are transformed into each other (Tekin, 2014) depending on the current phase of the development process (Kehl, 2019, p. 92). If diverse structures used the same classification information, such transformations could be realized very efficiently. Kehl et al. describe in (Kehl et al., 2016) how structuring via user-specific views on product data can be derived in this way.

The availability of classification could thus enable many interesting new functions in product development, e.g.:

- The similarity-based retrieval of components (e.g. spare parts) based on similar or identical characteristics of different components,
- the mapping of one product structure to another, and
- the related derivation of user-specific views according to (Kehl et al., 2016).

Note that classification requirements may vary from one product to another or at different placement locations within a product. For example, it must be possible to differentiate for production processes whether a wheel is mounted on the left or right side of a vehicle. For another product, an additional distinction may be needed between front and rear wheels. However, these are all wheels that have the same characterizing properties. In (Kehl, 2019, pp. 218–220) Kehl suggests to separate of context-neutral (wheel) from context-specific (right front wheel) classification information to model this circumstance. Since complex products such as vehicles potentially consist of many thousands of components (Stark, 2015, p. 6) and the amount of data created during development is ever increasing (Kehl et al., 2015), to automate the classification of existing components is necessary.

Based on a review of the literature (see Section 2) and to the best of our knowledge, automatically classifying product data in this manner, by making use of the metadata currently available in product structures, has not been studied before.

Thus, the novel contribution of this paper is twofold: First, we propose a concept for the classification of components into existing classes. Thereby, we consider both context-neutral components and their context-specific characteristics to support the aforementioned use cases.

Our concept relies on automated classification of product data using supervised machine learning techniques. As training data, we expect product structures, using the parent node of a component as its class, and its meta-data as features. The number of classes is consequently solely limited to the product structure.

After elaborating on the industrial motivation underlying this work, Section 2 also provides an analysis of classification approaches from industrial practice and scientific literature. In Section 3, we describe a concept for automated classification based on machine learning, and evaluate this concept using a small case study with a realistic example dataset with 1440 samples. Within the evaluation, we evaluate and compare the performance of four machine learning techniques. Section 4 summarizes our findings and discusses future research venues.

2. User story at Volkswagen AG

A recently installed Volkswagen AG group project "virtual high rack" aims to establish a classification standard and a group-wide description schema for all product components. This classification will be used to find and reuse components by property-based searches. The initialization of a virtual high rack is structured as follows: (i) definition of classes, (ii) determination of the properties of the respective classes, (iii) specification of product components as described in (Kehl et al., 2015), and (iv) assignment of components to classes.

Especially the assignment of components to classes is quite difficult to do manually, due to (i) the number of components to be classified, (ii) the spreading across different domains, brands and responsibilities, and (iii) the resulting differing semantics.

In this paper, we assume a *known set of classes* with their respective properties. Furthermore, we expect that the *product components are described*. Therefore, this paper focuses on *the automated assignment* of *existing components* to *existing classes*.

2.1. Use of classification in industrial practice

The findings discussed in the following are based on a case study carried out at Volkswagen AG as part of the efforts mentioned in Section 2. We identified three use cases involving artifact types that use classification. These artifacts, their application of classification, and the gaps regarding the respective use cases mentioned in the introduction are described in the following.

https://doi.org/10.1017/dsd.2020.139 Published online by Cambridge University Press

2.1.1. Part number

The *part number* is a unique identifier for each component. In the study at hand, it was firstly introduced to support logistic processes within a company. In order to determine e.g. similar components as spare parts in the context of such logistic processes, each part number contains an alphanumeric reference to a so-called *component family* (class). In the case study, a total of 160 different classes could be identified on the basis of this part number.

The main advantage of the part number is its group-wide availability for all components. However, since the part number is a historically grown identifier that carries meaning, the uniquely identifying value ranges per family are limited. In connection with the huge numbers of product variants (Eigner et al., 2011), the leading segments of the part number are not sufficient to identify all components of a component family. This shortage often leads to incorrect assignments of components to families and therefore to classes in day-to-day operations. Simply extending length and value ranges of the part number is not possible due to dependencies to many software systems. The problem intensifies with the continuing creation of new parts. In addition, only context-neutral components are classified with the part number. Classification against the background of a specific product (usage) is not supported.

2.1.2. Defect classification

The second classification use case we found is *defect classification*, i.e. classifying components on which errors occurred. Since errors usually occur during operation, various *context-specific* information is recorded during error analysis for each affected component. As this classification exists only for components that have caused errors, it is solely available for a small proportion of all components. In contrast to the part number however, context-specific information is recorded within the defect classification.

2.1.3. Product structure

The most common way to store product data in practice is in tree-like component structures such as BOMs (Adolphy et al., 2015; Kehl et al., 2016). Children of a node within such structures often give information about a further decomposition of the node into smaller units, e.g., car → front-end → drivetrain → engines. The actual components (engines) are subordinated to the lowest nodes (leaves) in a BOM. The assignment of components to nodes has an effect similar to classification (Schürr et al., 2008). Thus, 2600 classes are distinguished in the largest structure analyzed in the case study. The hierarchical structure is relatively stable and changes rarely. However, assigning a component to a node is a manual process and therefore potentially error-prone. Additionally, the classification created in this way can be ambiguous, since, as mentioned above, a structure also contains decomposition information or simple collectors that do not really represent a class in the original definition.

2.2. Related work

This section discusses related scientific work. Relevant papers are grouped according to the use cases (i) finding components, (ii) deriving user-specific views, and (iii) transforming product structures (see Section 1).

2.2.1. Finding components

In product lifecycle management (PLM), classification can either be used to classify products or their components. The first is a vivid research area; a good starting point might be (Fan et al., 2015). The second can be divided again in context-specific classification, such as in a BOM, and classification systems, which are used to classify context-neutral components (Eigner and Stelzer, 2009, p. 71). In (Yiu Ip and Regli, 2005), Yiu Ip et al. focus on the classification of context-neutral components by applying machine learning techniques. Therefore, they classify components based on their geometry (Yiu Ip and Regli, 2005).

In contrast to Fan et al. in (Fan et al., 2015) who classify products, we concentrate on the classification of components. While Yiu Ip et al. in (Yiu Ip and Regli, 2005) concentrate on classifying context-

https://doi.org/10.1017/dsd.2020.139 Published online by Cambridge University Press

neutral components, we additionally look into context-specific components. Moreover, in contrast to (Yiu Ip and Regli, 2005), we use the metadata instead of using the geometries directly.

2.2.2. User-specific views

A further tool for organizing the data in PLM are various product structures (Eigner and Stelzer, 2009, p. 78). In general, the costs for creating and managing product structures in relation to the overall cost of product development is relatively high (Adolphy et al., 2015). Moreover, as the complexity of products such as automobiles becomes more complex (Culler and Anderson, 2016), increasing effort for handling product structures can be assumed. Since product structures are an essential ingredient of product data management (Adolphy et al., 2015), the problem of their increasing cost is highly relevant. Besides that, in (Culler and Anderson, 2016) Culler et al. recommend introducing of filters on product data to further customize software used in PLM. These filters are similar to the user-specific views described in this paper. Another concept for such views is proposed in (Kehl et al., 2016), who derive views based on previously classified product data. They are the first who propose deriving user-specific views from products by using classes. Additionally, in (Adolphy et al., 2015), Adolphy et al. apply clustering to provide product structures based on groups of similar usage.
As recommended in (Culler and Anderson, 2016) and similar to (Adolphy et al., 2015) as well as (Kehl et al., 2016), this work further individualizes software used in PLM by creating user-specific views. In contrast to (Kehl et al., 2016), this work does not assume that product data is already classified, but rather creates the classification. Therefore, nevertheless, some classified training data is necessary. While Adolphy et al. (2015) use unsupervised learning, our approach uses supervised learning, because the case study showed that previous product structures can be used as training data.

2.2.3. Transforming product structure

Several works are based on the existence of high-quality product structures. One example for that is the one from (Morshedzadeh et al., 2019), who propose a provenance system to connect information created during the product lifecycle better to previously existing data. In 2005, Eigner et al. pointed out the topic of connecting different product structures (Eigner and Stelzer, 2009, p. 79). More recently, in (Huber and Sendler, 2013) Huber states that connections between product data at different phases of PLM are still not complete and managed (if at all) via heterogeneous interfaces. The process of publishing product data changes through these often interferes with data consistency during the PDP, because the interfaces are mostly constructed only for one specific transformation (Huber and Sendler, 2013).
In (Adolphy et al., 2015), the authors observe that existing processes for creating and transforming product structures are highly manual and error-prone. In this paper, we investigate feasibility of an automated process, which can reduce the number of interfaces and increase data consistency and quality during the PDP. (Morshedzadeh et al. (2019) also try to connect product data, but there work about including additional information in virtual models is based on the product structures we want to transform.

2.3. Research gap

For managing the huge amounts of data created during the PDP (Stark, 2015, p. 6) and connecting such, which is essential for future developments of PLM such as digital twin (Tao et al., 2018), classification systems are used to support users in retrieving components (Eigner and Stelzer, 2009, p. 76). The necessity of having different views on the product data within the PDP has been stated by (Adolphy et al., 2015; Culler and Anderson, 2016; Kehl et al., 2016). In addition, integrating different product structures along the PDP is essential for a consistent development of products (Huber and Sendler, 2013; Eigner et al., 2005). The previously explored classification approaches and industrial practice (see Section 2) show that researching classification to fulfil the use cases described above is promising. Despite the flood of product data created during the PDP (see Section 1), to the best of our knowledge there is currently no concept to classify product data automatically, consistent along the PDP, and cross-product to accomplish the identified use cases. The work presented in this paper towards a method for automating the classification task is a step towards filling this gap.

3. Machine learning for classifying product data

In comparison to humans, trained machine learning models are able to interpret large numbers of attributes in short time. This is one reason why we transfer the problem of assigning classes to components into a classical machine learning problem: If one considers the product data elements D in a PLM backbone system as components or elements with attributes, the problem of assigning a class c to each to each element in D can be understood as a classification problem (see Equation 1):

$$f(d) = \mathrm{c}\ \forall\ \mathrm{d} \in \mathrm{D}, \mathrm{c} \in \mathrm{C} \quad (1)$$

3.1. Training data

In the case of supervised machine learning, training data is needed to learn the described function f. The sample dataset used for the experiments reported in this section consists of 1440 samples, each describing the width, length and depth in millimeters of a brake pad for disk brakes of a car. Each of the elements belongs either to the front or back axle. To handle missing values, we firstly remove entries with more than one missing value. Secondly, we apply the common multivariate imputation by chained equations procedure (Azur et al., 2011) to replace missing data.

A scatterplot of the sample dataset is shown in 0. Due to different brake loads, a brake pad can either be assembled at the front axle, first class, or back axle, second class. The goal in this example is to automatically determine on which axle a brake pad has to be assembled.

The complete source code of our system including the sample data, extracted from an online shop for car spare parts[1], is available on GitHub[2]. We assume that the data is correct, but because we have collected the data online, we cannot verify this.

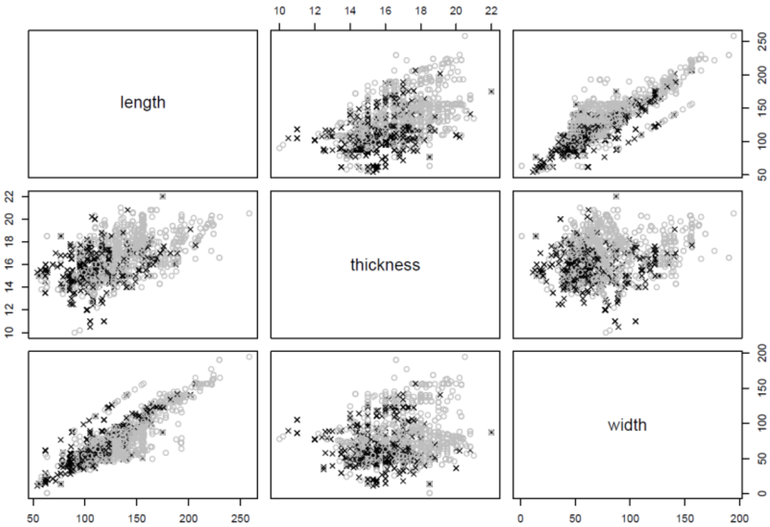

Figure 1. Dataset of brake pads for disc brakes of a car with attributes (length, thickness and width in mm) and location (gray circles for front axle, black crosses for rear axle)

3.2. Machine learning method

First, the dataset is randomly separated into a training and a test dataset. The probability of a sample being used in the test dataset is 20 percent (5-fold cross-validation). To make the later results independent from the concrete division into training and test dataset, 5-fold cross-validation is repeated, leading to 100 dataset pairs.

Then, we examine four machine learning methods: (i) k-nearest neighbor algorithm (kNN) (ii) logistic regression, (iii) random forest, and (iv) neural network. These methods were chosen for the following reasons: A big advantage of kNN, which is one of the most common methods in machine learning

[1] http://www.pkwteile.de

[2] https://github.com/SorenSc/ClassificationOfProductData

https://doi.org/10.1017/dsd.2020.139 Published online by Cambridge University Press

(Wu et al., 2008), is that it is simple to use, as only a single parameter (the number of classes k) has to be determined to apply the technique. Logistic regression has similar characteristics; its sole parameter is the limit for the differentiation of classes τ. Random forest on the other hand is a more sophisticated *bagging technique*, which means multiple models (decision trees) are learned and their results are combined. Moreover, random forest is able to deal with heterogeneous data (Louppe, 2014, p. 26) and missing values (Louppe, 2014, p. 80). Neural networks, on the other hand,are a popular technique, particularly able to deal with large amounts of data, but more difficult to configure.

The parameters are set by applying cross-validation and are $k = 4$ for kNN and $\tau = 0.5$ for logistic regression. For random forest, the number of decision trees is 50 and up to two attributes are considered per node of a decision tree. We build a relatively simple feed-forward neural network with five layers, using a combination of multiple dense layers and a dropout layer.

Next, we train models for each of the four techniques, its parameters, and the 100 datasets mentioned above. Then, we apply the trained models to the corresponding test data. To compare the results of the techniques, we built the confusion matrix for each of the 100 datasets. Based on that, the minimal, average and maximal accuracy are calculated and shown in Table 1. In order to verify assumptions about comparisons of the techniques applied, we use a Friedman-test as recommended by (Derrac et al., 2011). To cover all common significance levels, the level is set to 0.001 and compared with the p-value p. The results show that kNN (0.8447 on average) and random forest (0.8517) classify more elements correctly than logistic regression (0.7270) and the neural network model (0.6581) with $p < 0.001$. Logistic regression also outperforms the neural network with $p < 0.001$.

Table 1. Accuracy of the classification of the brake pad sample dataset calculated over 100 train and test dataset pairs

Accuracy in percent	minimum	average	maximum
kNN	0.7804	**0.8447**	0.8980
logistic regression	0.6314	0.7270	0.7843
random forest	0.7922	**0.8517**	0.8941
neural network	0.5177	0.6581	0.7804

3.3. Discussion of results

The best of the machine learning methods we investigated, classifies 85 percent of the input product data are correctly, which does not appear to be a good result; we assume that classification accuracy should be definitely be higher than 90% for the three application use cases described in Section 2.1. However, in practice, the dataset to train the model would be much larger than the example dataset used in Section 3.2, which will also most certainly increase the accuracy of the techniques. Yet, we conclude concrete classification for all product data is difficult. The results shown in Table 1 do not lend themselves to specific judgments regarding the concrete methods used, but can deliver some more general insights: Model-based machine learning methods such as random forest and logistic regression perform better than instance-based machine learning methods, because the former do not directly access historical training data for the classification. The tendency to use ensemble learning (Domingos, 2012) seems to be transferable to the application of product data and may further improve classification quality. Neural networks do not achieve good results in our case study experiment, but their accuracy is highly likely to increase substantially with larger training datasets.

We showed by use of imputation that handling missing values in the training data is possible. Nevertheless, the concrete procedure depends on the distribution of missing values. For dealing with classification errors, different approaches like adjustment by users in discovering or using classification, e.g. as a recommender system for the users have to be evaluated in the future. The developed classification concept is able to react to changes such as the need for new classes. These changes come along with a corresponding adaptation of the data pre-processing procedure. To react to changes of the attribute values of product data entries, a new model can be trained after a certain number of changes occurred.

https://doi.org/10.1017/dsd.2020.139 Published online by Cambridge University Press

Comparing computation times for large datasets shows that kNN has the highest time complexity of the machine learning techniques we have investigated (Maillo et al., 2017; Komarek, 2004, p. 89; Louppe, 2014, p. 96). It can be said that all four machine learning techniques considered are able to classify 100000 or more product data entries.

4. Conclusion and future work

We presented a concept for automated assignment of components to predefined classes or classification of product data based on given product structures to support the following three use cases: (i) The similarity-based retrieval of components, (ii) transferring from one product structure to another and (iii) the related derivation of user-specific views according to (Kehl et al., 2016).

4.1. Usable product data

Inspecting existing product data in practice has shown that the metadata, especially in product structures, can be used as a basis for automated classification. Though some data is error-prone, the huge amount of usable data that is already organized in product structures is remarkable. As in many other areas of machine learning (Domingos, 2012), the case study also showed that the effort required for pre-processing the product data is high.

4.2. Machine learning

Our results so far do not support any general statement about the specific machine learning technique which would be best for our domain. Nevertheless, applying instance-based methods does not seem promising, due to the amount of data used in PLM. Instead, ensemble-learning techniques such as random forest appear to be a reasonable choice, as they also will allow us to deal with the increasing amount of data to be expected in the future. Ultimately, machine learning alone will not suffice; equally important are high-quality training data, on the basis of which machine learning models can be trained.

4.3. Limitations

Besides the ability to classify product data via machine learning, the success of implementing the proposed concept also depends on other factors, which we have not considered in this work so far. For instance, the case study shows that success of a specific classification system strongly depends on its user acceptance. In addition to that, classifying product data to simplify the retrieval of suitable components can conflict with concerns of roles and rights management, which may limit the components visible to a specific user.

4.4. Essential properties of the developed concept

Firstly, the classification of the developed concept can be used during the whole lifecycle of components and thereby, supports the design process of products. Moreover, our concept is able to react to changes such as changing the attributes considered at the classification. Therefore, only the data preparation process has to be adopted and a new machine learning model can be trained.

4.5. Outlook

One auspicious approach for dealing with misclassified product data is interactive machine learning integrating human feedback into a machine learning model. Future work should also specify dealing with misclassified data further. For increasing the classification accuracy, machine learning techniques, which use boosting, such as XGBoost, seem to be promising; others, such as neural networks, should be reconsidered when real product data is available in a larger amount. Moreover, quantitatively verifying the automated approach presented in this paper by investigating the present manual process further is necessary. In addition, as mentioned before, we shall experiment with larger datasets and optimize our classification algorithms correspondingly. To study user acceptance and explore tradeoffs between accuracy, precision, and recall, a prototypical recommender application will be created that proposes similar components to expert users.

References

Adolphy, S. et al. (2015), “Method for Automated Structuring of Product Data and its Applications”, *Procedia CIRP*, Vol. 38, pp. 153-158.

Azur, M.J. et al. (2011), “Multiple Imputation by Chained Equations: What Is It and How Does It Work?”, *International Journal of Methods in Psychiatric Research*, Vol. 20 No. 1, pp. 40-49.

Culler, D.E. and Anderson, N.D. (2016), “A Paradigm Shift towards Personalized and Scalable Product Development and Lifecycle Management Systems in the Aerospace Industry”, *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, Vol. 10 No. 4, pp. 691-699.

Derrac, J. et al. (2011), “A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms”, *Swarm and Evolutionary Computation*, Vol. 1 No. 1, pp. 3-18.

Domingos, P. (2012), “A few useful things to know about machine learning”, *Communications of the ACM*, Vol. 55 No. 10, p. 78.

Eigner, M., Hauff, M.V. and Schäfer, P.D. (2011), “Sustainable Product Lifecycle Management: A Lifecycle based Conception of Monitoring a Sustainable Product Development”, In: Hesselbach, J. and Herrmann, C. (Eds.), *Glocalized Solutions for Sustainability in Manufacturing*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 501-506.

Eigner, M. and Stelzer, R. (2009), *Product Lifecycle Management: Ein Leitfaden für Product Development und Life Cycle Management, VDI, 2., neu bearb. Aufl.*, Springer, Dordrecht.

Eigner, M., Weidlich, R. and Zagel, M. (2005), *The Conceptual The Conceptual Product Structure as Backbone of the Early Product Development Process, Science Days 2005*, Darmstadt.

Fan, S., Lau, R.Y.K. and Zhao, J.L. (2015), “Demystifying Big Data Analytics for Business Intelligence Through the Lens of Marketing Mix”, *Big Data Research*, Vol. 2 No. 1, pp. 28-32.

Huber, A.S. and Sendler, U. (2013), “Das Ziel Digital Enterprise: die professionelle digitale Abbildung von Produktentwicklung und Produktion”, In: *Industrie 4.0, Xpert.press*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 111-124.

Kehl, S. (2019), *Marken- und domänenübergreifendes Management industrieller Produktdaten*, Springer Fachmedien Wiesbaden, Wiesbaden.

Kehl, S. et al. (2016), “Static Product Structures: An Industrial Standard on the Wane”, In: Harik, R., Rivest, L., Bernard, A., Eynard, B. and Bouras, A. (Eds.), *Product Lifecycle Management for Digital Transformation of Industries*, Springer International Publishing, Cham, pp. 69-78.

Kehl, S., Stiefel, P. and Müller, J.P. (2015). “Changes on Changes: Towards an agent-based approach for managing complexity in decentralized product development”.

Komarek, P. (2004), *Logistic Regression for Data Mining and High-Dimensional Classification*, Pittsburgh, PA.

Louppe, G. (2014). “Understanding random forests: From theory to practice”, *arXiv preprint arXiv:1407.7502*.

Maillo, J. et al. (2017), “Exact fuzzy k-nearest neighbor classification for big datasets”, *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, Naples, Italy, 09.07.2017 - 12.07.2017, IEEE, pp. 1-6.

Morshedzadeh, I., Ng, A.H.C. and Amouzgar, K. (2019), “Management of Virtual Models with Provenance Information in the Context of Product Lifecycle Management. Industrial Case Studies”, In: Stark, J. (Ed.), *Product Lifecycle Management (Volume 4): The Case Studies, Decision Engineering*, Vol. 16, Springer International Publishing, Cham, pp. 153-170.

Schürr, A., Nagl, M. and Zündorf, A. (2008), *Applications of Graph Transformations with Industrial Relevance*, Vol. 5088, Springer Berlin Heidelberg, Berlin, Heidelberg.

Stark, J. (2015), *Product Lifecycle Management*, Springer International Publishing, Cham.

Tao, F. et al. (2018), “Digital twin-driven product design, manufacturing and service with big data”, *The International Journal of Advanced Manufacturing Technology*, Vol. 94 No. 9-12, pp. 3563-3576.

Tekin, E. (2014), “A Method for Traceability and “As-built Product Structure” in Aerospace Industry”, *Procedia CIRP*, Vol. 17, pp. 351-355.

Wu, X. et al. (2008), “Top 10 algorithms in data mining”, *Knowledge and Information Systems*, Vol. 14 No. 1, pp. 1-37.

Yiu Ip, C. and Regli, W.C. (2005), “Content-Based Classification of CAD Models with Supervised Learning”, *Computer-Aided Design and Applications*, Vol. 2 No. 5, pp. 609-617.

https://doi.org/10.1017/dsd.2020.139 Published online by Cambridge University Press