A semantic web approach for structuring data-driven methods in the product development process

Benjamin Gerschütz[1]*, Benjamin Schleich[1], Sandro Wartzack[1]

[1] Engineering Design, Friedrich-Alexander-University Erlangen-Nuernberg

** Korrespondierender Autor:*
Benjamin Gerschütz
Martensstraße 8
91058 Erlangen
Telefon: +49 (0)9131/85-23659
Mail: gerschuetz@mfk.fau.de

Abstract

For the successful integration of data-driven methods in existing processes, it is first necessary to know these methods and their fields of application, potentials, and individual limits. This requires an easily accessible knowledge base. Depending on possible use cases and available data, this knowledge base should suggest methods that can lead to an optimization of the overall product development process. To prevent the knowledge base from being a rigid catalogue, it is essential to link it to the existing methods of the product development process. This contribution presents an approach based on the semantic web application Semantic MediaWiki, that ensures a connected representation of method knowledge and at the same time enables a link with use cases in the product development process.

Keywords

digital engineering, data-driven-method, design-process, semantic-web

 | DOI: https://doi.org/10.35199/dfx2021.15

1. Motivation

According to German initiatives, the integration of artificial intelligence (AI) in processes of small and medium-sized companies (SMEs) offers great potential [1]. Nevertheless, such an integration is not present in the development processes of those companies [2], although many of them identified artificial intelligence as a crucial precondition for their future success [3].

The central idea of many AI methods is the use of existent data to get an insight into unknown correlations. In engineering design, efforts were made to take advantage of the new methods. Some of them will be explained later in the contribution. The overall potential of those data-driven methods in engineering design has been shown earlier [4]. Nevertheless, some challenges have to be solved when integrating those methods in daily industrial use [4].

Before companies can analyze whether their data pool suits their problem or whether new methods can be integrated into the company structure, they must know details about available methods, their potentials, weaknesses, and prerequisites [5]. Unfortunately, most SMEs lack that knowledge, since no specialist employees are available. This is shown, for example, by the trend in vacancies with the job title "Data Scientist" for Great Britain. It can be seen that the frequency of job advertisements and thus the demand for new workers has been growing continuously since 2011 [6]. Current literature sources and method catalogues cannot solve this problem satisfactorily, since the knowledge is only provided separated, without contextual information to product development, e.g. WITTEN et al. [7]. There are some overview publications like those from BERTONI [8] or SHABESTARI et al. [9], but they only analyze certain parts of the whole process. Additionally, their findings are only valid for a short time and no real classification of the methods due to their applicability in the process development process is performed.

For the successful introduction of data-driven methods, an easily accessible knowledge base is beneficial. Depending on possible use cases and available data, this should suggest methods that can lead to an optimization of the overall process. For this, a linkage of the methods with the product development process is necessary.

This leads to the central research question of the paper:

- How can data-driven methods be captured in a targeted manner concerning product development?

The central goal of the contribution is to develop an easily accessible, easy-to-understand platform to provide context-sensitive knowledge about data-driven methods concerning applications in product development processes. Potential aspects are available data, necessary process outputs, and best practice use-cases.

The remaining contribution is structured as followed. In Section 2, the theoretical background about data-driven methods and semantic web applications are clarified and definitions about some central keywords were given. In section 3 the used methods and tools are being specified. The developed approach for semantic and structured knowledge representation is presented in section 4. In Section 5 a critical analysis of the findings is done, and further research approaches are identified.

2. State of the Art

In the following, a systematic approach for the preparation of knowledge about data-driven methods is introduced.

2.1. Data-driven Methods

In general, a data-driven method supports decisions based on data or even makes autonomous decisions [10]. The overall context of data-driven methods and tools is difficult to grasp as many methods such as k-nearest neighbour can be used for different use cases (regression, classification), but with different objectives. Additionally, data mining and machine learning lack a sharp distinction and are used synonymously sometimes, but also with different meanings [7]. The common definition of data mining is given by FAYYAD, PEATESKY-SHAPIRO and SMYTH [11] describing data mining as the extraction of patterns from data by the application of specific algorithms.

For machine learning, some different definitions exist. The most universal is given by SAMUEL [12], who defines it as a field of study, giving computers the ability to learn a certain task without being explicitly programmed to do this. In the context of product development, machine learning is often used in terms of knowledge extraction and decision-making [9]. One possible goal of machine learning algorithms in product development processes is the generation of predictive meta-models [13].

Data mining and machine learning enable the processing of a wide variety of tasks with a wide variety of objectives. The most common ones are briefly outlined below. Classification aims to divide objects into two or more classes with the help of already known, similar objects [14]. The classification of the objects is done by previously defined rules. Common classification methods include support vector machine classifiers, naïve Bayes classifiers, k-nearest neighbour methods, and decision trees [7]. In the context of data mining, the boundary between classification and regression methods is blurred, so that a large part of the classification methods are extended to regressive models [15]. The core concept of regression is to model correlations between different characteristics based on a database. By examining the function modelled in the process, these feature correlations can be derived and controlled [14]. An advantage of regression methods is that the strength of the respective correlations can also be derived, and it is possible to make predictions about missing feature values [16].

Supervised learning, a subarea of machine learning, is the development of forecast models that are previously trained by known data. By matching correct output and predicted output, forecast errors can be calculated. By reducing the forecast errors, the model can be optimized. In addition, the precision of a model increases as the amount of training data increases. [7] Most regression and classification methods can also be used for supervised learning [17]. Like supervised learning, unsupervised learning is also part of machine learning. Here, neither known target values are available as a basis, nor is a reward system used [14]. Since, in contrast to supervised learning, no concrete output data are available, an attempt is made instead to recognize patterns and structures in these data using the feature values of the input data [7]. In addition to unsupervised and supervised learning, reinforcement learning is another major subfield of machine learning. In reinforcement learning, a so-called agent attempts to learn independently, with the help of a reward system, which actions are required in a certain situation to maximize the reward [18].

2.2. Semantic Web

Due to the highly interconnected structure of information and the fast-moving nature of the overall context, classic methods of processing and documenting knowledge, such as rigid databases, quickly reach their limits. Semantic web solutions enable computers to understand the meaning of documents and data by providing further information or definitions [19].

Common terminology is key for that kind of application [20]. During the conceptualization, a vocabulary with classes, relations, and instances has to be defined [21]. These offer the possibility to capture knowledge in a network and make it available for machine evaluation. The links between individual knowledge elements can be given a meaning that enables context-sensitive evaluation. Thus, a knowledge element "A" can be a potential of one method and at the same time a limit of another method. A simple link to the knowledge element "A" in the method description establishes a connection between the individual elements, but only the semantic information "is_potential" or "is_boundary" gives the connection a meaning and converts the stored information into knowledge.

OCKER et al. [22] use this approach, to link production knowledge in early design phases. Furthermore, consistent requirements engineering processes can be efficiently and purposefully supported by ontologies [23] and tolerancing processes can be automated [24].

3. Methods and Tools

To answer the aforementioned research question, existing methods and tools were analyzed in the context of data-driven methods. The focus was on links between the individual components. These links were then transferred into a semantic data model and implemented in a semantic wiki. The relevant information was collected as part of a literature study.

To realize the knowledge base in a semantic wiki, an instance of Semantic MediaWiki (SMW) [25] was used. SMW is a free and open-source extension of MediaWiki [26], which itself forms the basis of the well-known Wikipedia. MediaWiki itself only provides basic wiki tools like page hosting and editing as well as common hyperlinks in between. SMW empowers those capabilities by adding semantic functionality giving those links a meaning and the whole system the option to export and analyze the stored knowledge.

4. Results

Based on the state of the art of data-driven methods, we have developed a “class model of data-driven methods”, which we will introduce in the next subsection. Since the concept of class models is rarely known in product development, a short introduction is given here.

A class diagram is mostly based on the syntax of the unified modelling language (UML) and gives the structure, behaviour, and interfaces of objects, mostly in software development [27]. Since we do not need the whole UML specification of class diagrams, only some of the aspects are explained. In general, classes are depicted as a rectangle with the name of the class on top. Additionally, attributes, operations, and properties can be specified and are separated by horizontal lines.

To clarify the use of a class diagram, Figure 1 shows the class diagram of a roller bearing. It consists of an inner and an outer ring with balls in between them. All elements are of a certain material and the whole bearing is mounted on a shaft. Up in the centre is the main class *rollerBearing*. The class has the attributes *innerDiameter* and *outerDiameter,* both as an integer. Additionally, the roller Bearing has several Balls. The corresponding class is placed below the *rollerBearing* class. The small Numbers on the connection represent the cardinality, meaning how many balls are in one roller bearing. Since every ball can only be in one bearing the number on this site is 1, the number on the ball class side is n, meaning it is not exactly defined how many balls are in one bearing. A similar representation is found in the connection to the class *ring.* Here the cardinality is “0…2” meaning a bearing can be realized with everything between 0 and 2 rings. The classes *outerRing* and *innerRing* are of a special kind, represented by the white arrowhead. They are specializations of the *ring* class, which means that they have all the properties of the superclass and are suitably extended additionally. The *shaft* class has no direct connection to the *rollerBearing* but only a relation (the assembly). This is visualized through a dashed connection.

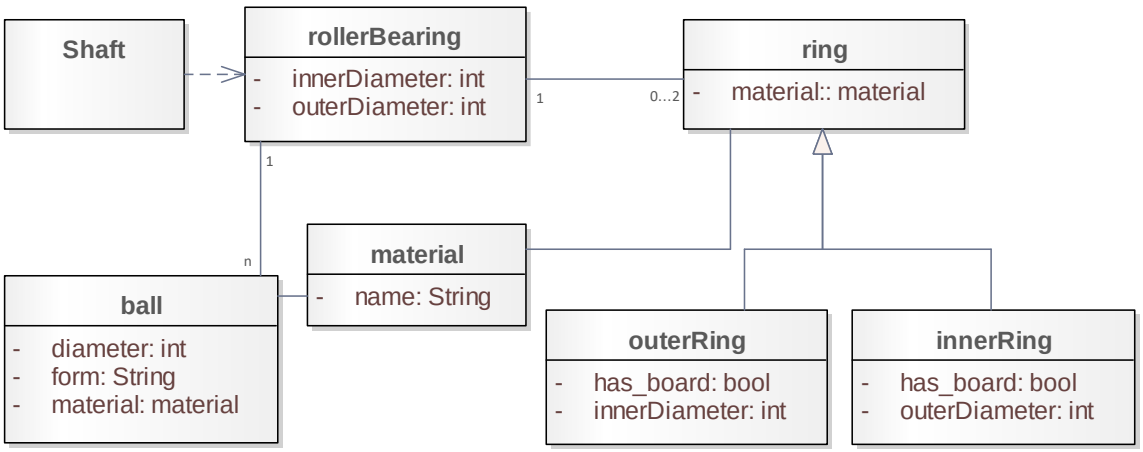

Figure 1: Class diagram of a roller bearing

4.1. Semantic Data Model

To generate a semantic model using the class diagram, the properties of the different classes and the connection between classes are translated to semantic properties. It is best practice to create those properties as verb phrases [25]. An example with the material of the presented roller bearing should make this practice more comprehensible. In the two terms – steel is the material of the outer ring. ←→ The outer ring material is steel – the property named "material" could be assigned, but this does not cover the exact knowledge and meaning behind it. In the first sentence, the better naming is "is_material_of" and in the second, it is "has_material" to define a direction of the meaning.

4.1.1. Overview

The central goal of the presented data structure is the structured capture of knowledge about data-driven methods based on semantic links. In Figure 2 the general structure of the data-driven methods class diagram is shown. It consists of four main classes, even if some properties could be further specified in additional classes. The individual classes are explained below and the entries are translated into semantic properties.

4.1.2. Class Category

The most general class is the *category* class. The purpose of this class is to give a first overall structure in the sense of namespaces. One key differentiator between individual instances is the capability of doing prognosis, realized by the bool-type property prognosis.

One member of the class *category* is data mining, which does no prognosis. Additionally, data mining is used by the concepts classification and regression and is realized by the method k-nearest neighbour. The class properties, the corresponding semantic properties, and the example is shown in Table 1.

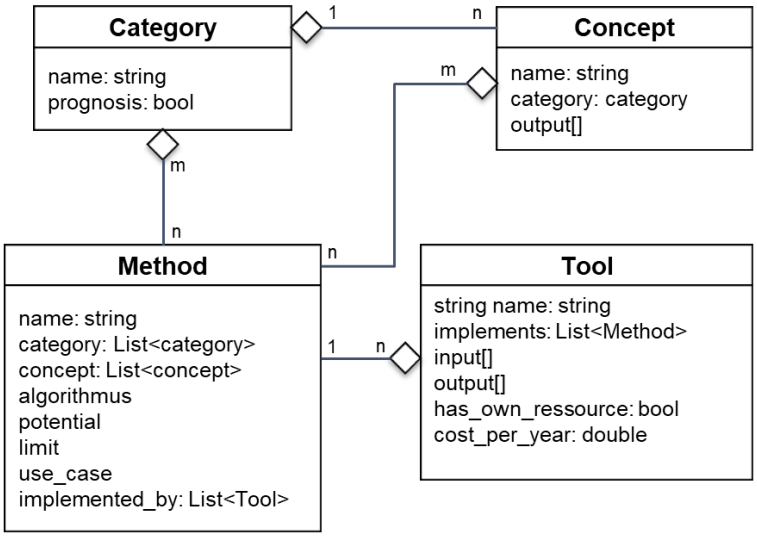

Figure 2: Class diagram of data-driven methods

Table 1: Semantic properties of the class *category*

class property	semantic property	Example
name	has_name	data mining
prognosis	does_prognosis	false
concept	used_by_concept	[classification, regression, …]
method	realized_by_method	[k-nearest neighbour, …]

4.1.3. Class Concept

To further specify the defined *category*, the class *concept* is introduced. In this class, a definition of the overall scope is done. Relevant properties of this class are the concept name, the category, this concept belongs to and the abstract output – in the sense of for example a class or a numerical prediction. Since the input is strongly dependent on the algorithm or tool used, it is not relevant here.

One example for an instance of this class is classification. It is a concept of category data mining and generates the abstract output of an object class. The concept is specified, among others, by the method k-nearest neighbour. The class properties, the corresponding semantic properties, and the example are summarized in Table 2.

Table 2: Semantic properties of the class *concept*

class property	semantic property	Example
name	has_name	classification
category	has_category	data mining
output[]	has_output	object class
method	specified_by_method	[k-nearest neighbour, …]

4.1.4. Class Method

The central class is the *method*. Here the link to use cases in the product development is done. Like in the classes introduced before, the link to the other classes is realized by the properties category and concept. Furthermore, the property algorithm is used for a description of the relevant algorithm used by the method. Since not all methods cover the same potentials

and limits, although they are specifying the same concept, the corresponding properties allow a distinction here. In the use_case property, potential use cases in product development and design are linked. The identification is done by literature review and industrial use-cases. The last property gives a link on Tools, which implements the method and thus makes it usable.

In Table 3 the presented properties are translated to their semantic counterparts. As an example, we analyze decision trees. The method is from the category of data mining and specifies the concept of classification [7]. An algorithm used to realize the method is the top-down induction of decision trees [28]. The method has the potential of a simple but efficient classification and decision making [29], but the user has to avoid overfitting and cannot apply the method if the goal is to model correlations [7]. One potential use case of decision trees is to perform a requirements analysis and support the inherent decision making (Quelle). Some implementations of decision trees exist. One of the most common implementations is the DecisionTree module in the Python-Package Scikit-Learn [30]. This module further splits into classification and regression classes, performing individual concepts.

Table 3: Semantic properties of the class method

class property	semantic property	Example
name	has_name	Decision Tree
category	has_category	data mining
concept	speciefies_concept	classification, regression
algorithm	uses_algorithm	Top-Down Induction of Decision Trees
potential	has_potential	[classification, decision making]
limit	has_limit	no correlation, overfitting
use_case	used_in	[requirements analysis, …]
Implemented by	Is_implemented_by	DecisionTreeClassifier class of Scikit Learn

4.1.5. Class Tool

Since elements of the class *method* only give abstract algorithms of the described methods, a *tool* is needed, to use those methods practically. It should be emphasized here that the tools in this class should only be used for one specific task at a time. For larger packages such as scikit-learn, reference should therefore be made to the relevant modules or subclasses. The overall toolbox is documented through the corresponding property. For example, for the presented decision trees, there are different classes within the package scikit-learn, which allow both classification and regression. Since these are different concepts, this is recorded separately. Some tools need training data to generate a model. If this input is needed, the property training input can record this. To represent product development, the input and output properties consider the typical data types of product development. Additional aspects cover the management part of method integration. Since not all methods need resources (e.g. calculation-servers) in the companies, a recording of the resource capabilities of the individual tools is done as well as a cost estimate. This gives the companies the option to do a preliminary risk analysis.

An example instance of this class is the scikit-learn DessionTreeClassifier [30]. The tool implements the decision tree method and uses the concept of classification. For training, the tool needs a numeric array of training samples and a numeric array of class labels. During usage, the input of this implementation is a numeric array, which is transformed into a class probability as output. Since scikit-learn is an open-source package, it has no resources and zero cost per year.

Table 4: Semantic properties of class Tool

class property	semantic property	Example
name	has_name	DecisionTreeClassifier
toolbox	has_toolbox	Scikit-learn
implements	Implements_method	Decision Tree
concept	use_concept	Classification
training input	needs_for_training	a numeric array of training samples and a numeric array of class labels
Input	Has_input	Numeric array
output	has_output	class probability of the input
has own ressource	has_ressource	no
cost per year	generate_cost	0

4.2. Demonstrator

To realize the shown concept, a demonstrator was implemented. The whole system works on a webserver with Mediawiki Version 1.35.2 [26] with the Addon Semantic MediaWiki (SMW) Version 3.2.3 [25]. The system supports the well-known wiki functionalities with collaboration capabilities and change history. Additionally, semantic annotations and queries can be done with SMW. Therefore, it is a powerful basis for a connected knowledge base of design process relevant method knowledge about data-driven methods and tools.

4.2.1. Implementation

The general use of SMW is like the use of Wikipedia. The semantic information is linked in the general text as well as in the information box on the upper right corner of each side.

Semantic links are realized by wiki links with relevant semantic keywords. A demonstrator page is shown in Figure 3. All entries in the information box have the structure *[[semantic_keyword::link_to_description_page]]*, therefore all information are summarised here. Additionally, in the continuous text, the links are registered as well.

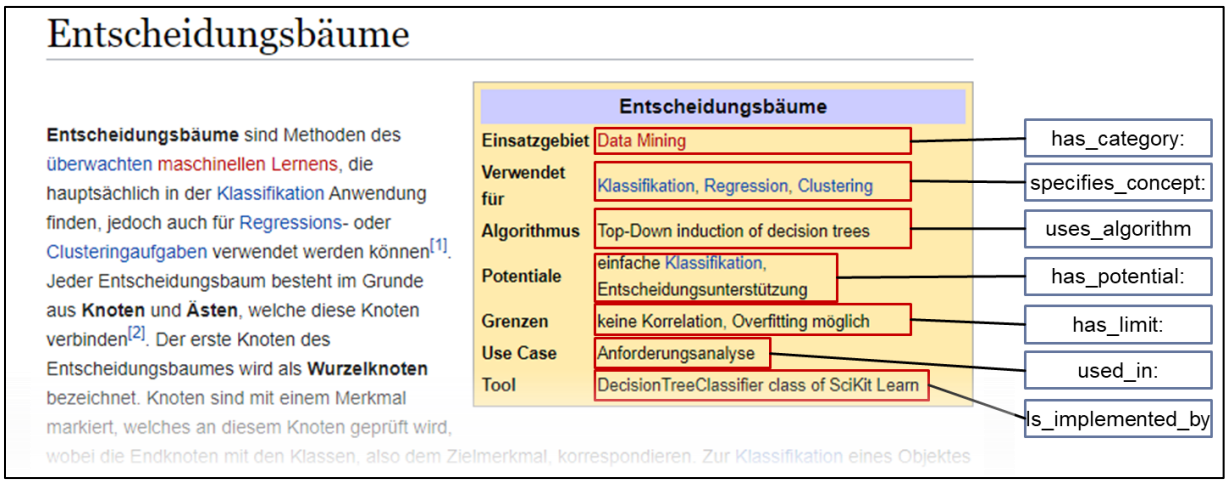

Figure 3: Semantic Wiki implementation

4.2.2. Knowledge Export

Recording knowledge as comprehensively and correctly as possible is only a first step. To obtain a powerful knowledge base, the stored knowledge must be extractable as easily and purposefully as possible. For this purpose, SMW also offers easy-to-use tools that allow knowledge extraction based on queries. Given the class instances from above, we want to find a method, which can perform a requirements analysis and can help to classify given requirements. Figure 4 shows the appropriate query on the left and the resulting methods on the right side. The query module enables more complex queries that take more than two aspects into account. With already defined boundary conditions, even clearer results are achievable.

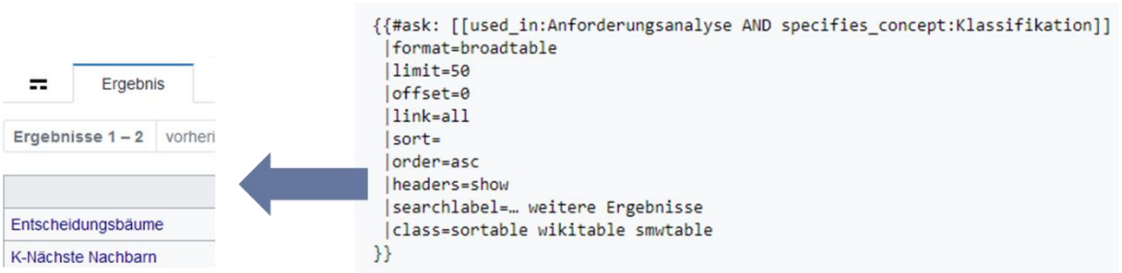

Figure 4: Example Query to perform a knowledge export.

5. Discussion

To integrate new methods in design processes, the companies need to know those methods, their potentials, weaknesses, and tools implementing the methods. To spread this knowledge, a powerful, online knowledge base is beneficial. To avoid building a “dead method catalogue”, the usage of semantic web technology seems a good idea. In this contribution, we presented a semantic model for data-driven methods and tools to realize a connected method knowledge base. Additionally, a demonstrator, based on Semantic MediaWiki was shown, realizing the class model in a proof of concept. The demonstrator shown is only a first step, which must be consistently filled with methods and use cases in the further course, some of which are also being developed within the framework of the FORCuDE@BEV research network and adapted to the drivetrain of electrified powertrains.

Further research will deal with an in-depth validation of the presented semantic. To do this in a targeted way, it is planned to develop the semantics into an ontology. Various methods are available for ontologies, which can be used to validate them and ensure that no incomplete models are created, or information is duplicated.

Acknowledgement

This research work is part of “FORCuDE@BEV - Bavarian research association for customized digital engineering for bavarian SME's“ and is funded by the “Bayerische Forschungsstiftung (BFS)”.

The authors are responsible for the content of this publication. Special thanks are directed to the Bayerische Forschungsstiftung (BFS) for financial support of the whole research project.

References

[1] Die Bundesregierung: Nationale KI Strategie (2018)

[2] Grimm, F.; Gentemann, L.: Digital Engineering - Agile Produktentwicklung in der deutschen Industrie : Bitkom Research, 2020

[3] item: Wie digital ist der Maschinenbau 2020?, 2020

[4] Gerschütz, B. et al.: Towards Customized Digital Engineering: Herausforderungen und Potentiale bei der Anpassung von Digital Engineering Methoden für den Produktentwicklungsprozess. In: Stuttgarter Symposium für Produktentwicklung 2021 (SSP 2021). Stuttgart, 2021

[5] Wuest, T.; Weimer, D.; Irgens, C.; Thoben, K.-D.: Machine learning in manufacturing: advantages, challenges, and applications. In: Production & Manufacturing Research Bd. 4, Taylor & Francis (2016), Nr. 1, S. 23–45

[6] van der Aalst, W. M. P.: Data Scientist: The Engineer of the Future. In: Mertins, K. ; Bénaben, F. ; Poler, R. ; Bourrières, J.-P. (Hrsg.): Enterprise Interoperability VI. Cham : Springer International Publishing, 2014

[7] Witten, I. H. ; Frank, E. ; Hall, M. A. ; Pal, C. J. (Hrsg.): Data mining: practical machine learning tools and techniques. Fourth Edition. Amsterdam : Elsevier, 2017

[8] Bertoni, A.: Data-driven design in concept development: systematic review and missed opportunities. In: Proceedings of the Design Society: DESIGN Conference Bd. 1 (2020), S. 101–110

[9] Shabestari, S. S.; Herzog, M.; Bender, B.: A Survey on the Applications of Machine Learning in the Early Phases of Product Development. In: Proceedings of the Design Society: International Conference on Engineering Design Bd. 1 (2019), Nr. 1, S. 2437–2446

[10] Montáns, F. J. ; Chinesta, F.; Gomez-Bombarelli, R.; Kutz, J. N.: Data-driven modeling and learning in science and engineering. In: Prof. Gomez-Bombarelli via Ye Li (2019). — Accepted: 2020-10-14T19:59:48Ztex.ids= montans2019publisher: Elsevier BV

[11] Fayyad, U.; Piatetsky-Shapiro, G.; Smyth, P.: From Data Mining to Knowledge Discovery in Databases. In: AI Magazine Bd. 17 (1996), Nr. 3, S. 37–37

[12] Samuel, A. L.: Some studies in machine learning using the game of checkers. In: IBM Journal of Research and Development Bd. 44 (2000), Nr. 1.2, S. 206–226

[13] Vajna, S. et al.: CAx für Ingenieure: eine praxisbezogene Einführung. 3., vollständig neu bearbeitete Auflage. Berlin, Germany : Springer Vieweg, 2018

[14] Mohri, M.; Rostamizadeh, A.; Talwalkar, A.: Foundations of machine learning, Adaptive computation and machine learning series. Cambridge, MA : MIT Press, 2012

[15] Beierle, C.; Kern-Isberner, G.: Methoden wissensbasierter Systeme. Wiesbaden : Springer Fachmedien Wiesbaden, 2014

[16] Sarstedt, M.; Mooi, E.: A concise guide to market research: the process, data, and methods using IBM SPSS statistics, Springer texts in business and economics. 2nd ed. Berlin Heidelberg : Springer, 2014

[17] Bonaccorso, G.: Mastering machine learning algorithms: expert techniques for implementing popular machine learning algorithms, fine-tuning your models, and understanding how they work, 2020

[18] Wiering, M.; Otterlo, M. van (Hrsg.): Reinforcement learning: state-of-the-art, Adaptation, learning, and optimization. Heidelberg ; New York: Springer, 2012

[19] Berners-Lee, T.; Hendler, J.; Lassila, O.: The Semantic Web. In: Scientific American Bd. 284 (2001), Nr. 5, S. 34–43

[20] Bernardi, A.; Holz, H.; Maus, H.; van Elst, L.: Komplexe Arbeitswelten in der Wissensgesellschaft. In: Pellegrini, T. ; Blumauer, A. (Hrsg.): Semantic Web: Wege zur vernetzten Wissensgesellschaft, X.media.press. Berlin, Heidelberg : Springer, 2006, S. 27–45

[21] Kügler, P.; Schleich, B.; Wartzack, S.: Consistent digitalization of engineering design – an ontology-based approach. In: Norddesign, 2018.

[22] Ocker, F.; Vogel-Heuser, B.; Paredis, C. J. J.: Applying Semantic Web Technologies to Provide Feasibility Feedback in Early Design Phases. In: Journal of Computing and Information Science in Engineering Bd. 19 (2019), Nr. 4, S. 041016

[23] Kestel, P. et al.: Ontology-based approach for the provision of simulation knowledge acquired by Data and Text Mining processes. In: Advanced Engineering Informatics Bd. 39 (2019), S. 292–305

[24] Goetz, S.; Schleich, B.: Ontology-based representation of tolerancing and design knowledge for an automated tolerance specification of product concepts. In: Procedia CIRP Bd. 92 (2020), S. 194–199

[25] semantic-mediawiki.org: Semantic MediaWiki. URL https://www.semantic-mediawiki.org/w/index.php?title=Semantic_MediaWiki&oldid=76616. - abgerufen am 2021-06-23

[26] MediaWiki: MediaWiki. URL https://www.mediawiki.org/w/index.php?title=MediaWiki&oldid=3878227. - abgerufen am 2021-06-23

[27] NORM: ISO/IEC 19505-2:2012 (2012)

[28] Rokach, L.; Maimon, O.: Top-down induction of decision trees classifiers - a survey. In: IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) Bd. 35 (2005), Nr. 4, S. 476–487

[29] Swain, Philip H.; Hauska, H.: The decision tree classifier: Design and potential. In: IEEE Transactions on Geoscience Electronics Bd. 15 (1977), Nr. 3, S. 142–147

[30] Pedregosa, F. et al.: Scikit-learn: Machine learning in Python. In: Journal of Machine Learning Research Bd. 12 (2011), S. 2825–2830.