Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 138 (2026) 821–826

www.elsevier.com/locate/procedia

18th CIRP Conference on Intelligent Computation in Manufacturing Engineering

Connection Feature Extraction in 3D CAD Assemblies using a Knowledge Graph

Tobias Köhler*, Nico Schmidt, Jan Martin Keil, Diana Peters

German Aerospace Center (DLR), Institute of Data Science, 07745 Jena, Germany

* Corresponding author. Tel.: +49 3641 30960 155; *E-mail address:* tobias.koehler@dlr.de

Abstract

Recycling is becoming increasingly important, due to growing resource shortages and the need of a reduction in CO_2 emission. Automated disassembly is essential for accelerating recycling processes and optimizing the extraction of valuable materials. To achieve an automated disassembly, it is crucial for an automated disassembly system to know the type of connections between components in an assembly. We present a method for extracting connections from 3D CAD models of assemblies. Our approach identifies connections through collision detection and populates a knowledge graph with the acquired data. This knowledge graph serves as foundation for software systems to assess product recyclability or to optimize recycling processes.

Peer-review under responsibility of the scientific committee of the 18th CIRP Conference on Intelligent Computation in Manufacturing Engineering (CIRP ICME ‘24)

Keywords: Knowledge Graph, Feature Technology, Ontology, Manufacturing, Connection Analysis, CAD

1. Introduction

In the 21st century, repairing and recycling of defective or outdated products play an important role in the preservation of resources and the protection of the environment. The recycling of products preserves valuable resources and enhances the sustainability of industrial processes as recycled materials can replace virgin materials. The increasing expectation on engineers to consider sustainability during the product design causes a growing need for automated solutions that support engineers in this task. Software algorithms capable of automatically analyzing assemblies are essential to provide reasoned decision support. By efficiently extracting connections and component data from 3D CAD (computer aided design) files, such algorithms facilitate the identification of recyclable designs and their optimal recycling routes, as the disassembly effort and the composition of component materials can be derived from this.

The current state-of-the-art solutions for feature extraction from STEP files are limited to the analysis of individual components [1], leaving a crucial gap in the analysis of assemblies.

Our goal is to fill this gap by developing a method that takes a 3D assembly STEP file as input, extracts all joints and stores them into a knowledge graph. For this, we need to perform two primary tasks: (1) model concepts of assemblies, components and their connections using a knowledge graph and (2) devise a robust approach for extracting connections from the structure of the STEP file.

The knowledge graph must describe the concepts of relationships between components as well as the relevant manufacturing processes to facilitate the derivation of corresponding separation processes. In our approach, the

Peer-review under responsibility of the scientific committee of the 18th CIRP Conference on Intelligent Computation in Manufacturing Engineering (CIRP ICME ‘24)
10.1016/j.procir.2026.01.141

knowledge graph is used to identify the types of connection on the one hand and to store the extracted data according to the corresponding concepts on the other.

The division of the problem into two distinct parts—extraction of connections and digital representation using a knowledge graph—enables a systematic and comprehensive solution of the challenges posed by 3D assembly analysis and processing.

We initially focus on the development and implementation of the method for connections of the three joining techniques screwing, riveting, and welding. Those joining techniques belong to three of nine joining process groups according to DIN 8589 [2] and represent all three types of connections (form fit, material fit, and force fit). Furthermore, both detachable and non-detachable joints are represented.

2. Related Work

The following sections provide an insight into the topics relevant to this work and the state of the art.

2.1. STEP–Standard for the Exchange of Product Model Data

Exchanging product information between different software tools in design and manufacturing is essential. Therefore, the Product Data Exchange Specification (PDES) was developed in the mid-1980s. This specification was submitted to the International Organization for Standardization (ISO) in 1988 and formed the basis for the ISO 10303 Standard for the Exchange of Product Model Data (STEP) [3,4]. This standard for the exchange of product model data has been continuously developed since then.

An essential part of STEP are the Application Protocols (APs) [5], which define the usage of STEP for a specific contexts. Two of the most commonly used application protocols are AP203 (Configuration controlled 3D design of mechanical parts and assemblies) [6] and AP214 (Core data for automotive mechanical design process) [7]. Since 2009, the two application protocols have been merged into a single protocol. The resulting AP242 covers a wide range of product development requirements [8]. STEP AP242 offers the possibility to annotate components. As an internationally recognized standard, STEP thus provides the necessary options for representing connections between individual parts.

2.2. Ontologies and Knowledge Graphs

According to Gruber, an ontology is defined as an "explicit formal specification of a common conceptualization" [9]. This specification comprises classes, properties and relationships between classes. Hence, specific knowledge domains can be represented and thereby enable machine-readability and the entailment of implicit knowledge.

Many formats have been developed for the representation of ontologies. One such format is the Web Ontology Language (OWL) [10], standardized by the World Wide Web Consortium (W3C). OWL ontologies are primarily exchanged as Resource Description Framework (RDF) [11] files, also standardized by the W3C. RDF employs triples, which consist of a subject, predicate, and object, to represent data. These triples can also be interpreted as a directed graph with edges and nodes, with the predicate forming the edge. That way, RDF can be used to create so called knowledge graphs.

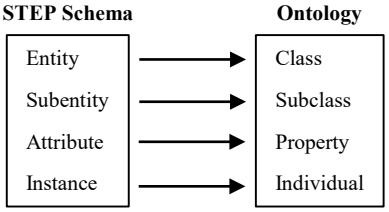

Figure 1: Mapping STEP schema into ontologies (according to [12])

In contrast to the STEP schema, entities in ontologies are referred to as classes. Consequently, subordinate classes are also called subclasses. The individual objects of a class are called individuals, and the properties of the classes or their individuals are referred to as properties. The terminology in STEP and the associated terms in relation to ontologies are summarized in Figure 9.

2.3. Separating Axis Theorem

The separating axis theorem can be employed to identify collisions between convex objects [13,14]. In this section we explain the procedure in case of two-dimensional objects and indicate how it can be extended to recognize collisions in case of three-dimensions objects.

The separating axis theorem postulates that two convex polygons will not collide with each other if they can be projected onto an axis so that the projections do not overlap. Such an axis is designated a separating axis [13].

In the most basic case, the polygons are two rectangles that are both aligned with the x and y axes. In this instance, the x and y axes are the potential separating axes. By determining the extreme values of the coordinates of the corner points of both rectangles, the projections onto the x and y axes can be established. The two objects only collide if none of the axes is a separating axis [13]. If the rectangles are not aligned with the same axis, there are four possible axes onto which the corner points of the rectangles must be projected.

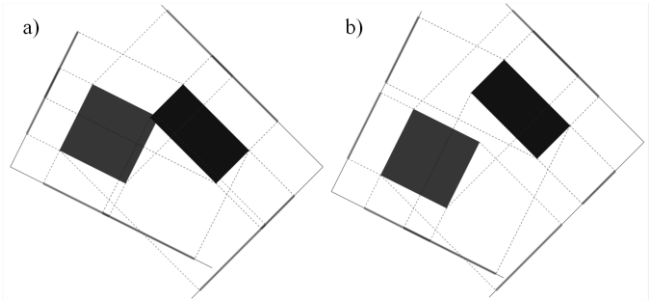

Figure 2: Illustration of Separating Axis Theorem; a) collision between objects; b) no collision between objects

Figure 2 illustrates the application of the separating axis theorem. It can be observed that there are two separating axes in Figure 2b, as the projections of the rectangles on these do not overlap. In Figure 2a there are no separating axes.

The described procedure can also be applied in the case of two three-dimensional boxes. However, in this case, 15 possible separating axes must be analyzed [13]. Six of the 15 axes in question result from the local coordinate systems of the two bounding boxes. The remaining nine axes are derived from the cross product of the axes of the two bounding boxes.

2.4. Feature Extraction

In the field of feature extraction from 3D CAD components, numerous approaches have been developed [1,15]. These include ontology-based methods, which are increasingly being pursued [12,16]. For example, Gupta and Gurumoorthy [17] present an algorithm that extracts shape features such as holes from STEP files and structures the gained knowledge in an ontology.

In a previous research [12], we followed an approach that analyses individual components based on concepts of geometric elements modelled in a knowledge graph. Afterwards, the extracted features are stored in this graph.

In STEP AP242 it is possible to annotate components, which can help to identify features, their properties or connections between components. To extract information, e.g. tolerances from 3D models, Mohammed et al. [17] propose to use the representation of shape and position tolerances and other production-relevant annotations in STEP format and examine welding in more detail as an application case.

Due to limitations in certain CAD programs' ability to export all annotations accurately, challenges arise in feature extraction. Furthermore, the current state of the art lacks knowledge-based approaches for extracting connections and connection types from 3D CAD models of assemblies.

3. Method

To fill this gaps in the state of the art, we present a method for extracting connections and connection types from 3D CAD models of assemblies. We discuss this method and its prototypical implementation in the following sections.

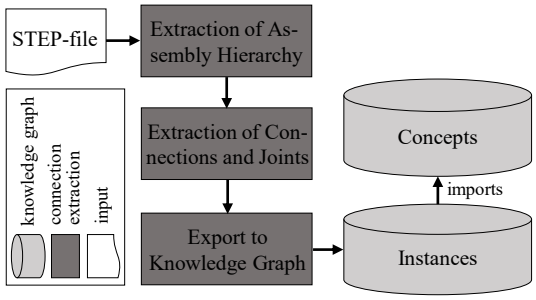

Figure 3: General approach of proposed method

The knowledge-based approach is divided into two main parts (1) data modelling in the knowledge graph and (2) connection extraction (see Figure 3). For the first part, we model concepts and relationships of assemblies and components in machine-readable form in a knowledge graph. Those concepts are one the one hand used for identifying the type of connection during the extraction process and on the other hand to be able to assign the extracted data as instances in the knowledge graph.

For the second part of the approach, we implement an algorithm that takes a STEP-file as input, extracts the connections and connection types from assemblies, and stores them as instances of the concepts in the knowledge graph. This connection extraction method is further divided into three main steps: (1) classification of components of the assembly and determination of hierarchy of components, (2) deriving of connections based on contact with joining components and (3) exporting of obtained information to the knowledge graph.

4. Knowledge Graph

In this section we introduce the data model of the knowledge graph. It comprises the three concepts assembly, component and connection type.

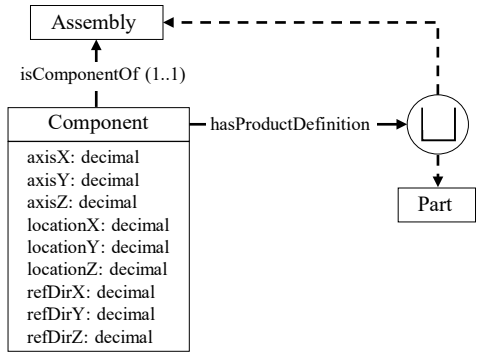

Figure 4: Data model of relationship between Assembly and Component

The *Assembly* class represents assemblies in CAD models. An assembly consist of components of the *Component* class (see Figure 4). A component is always installed in exactly one assembly, which is reflected by the cardinality of the *isComponentOf* property.

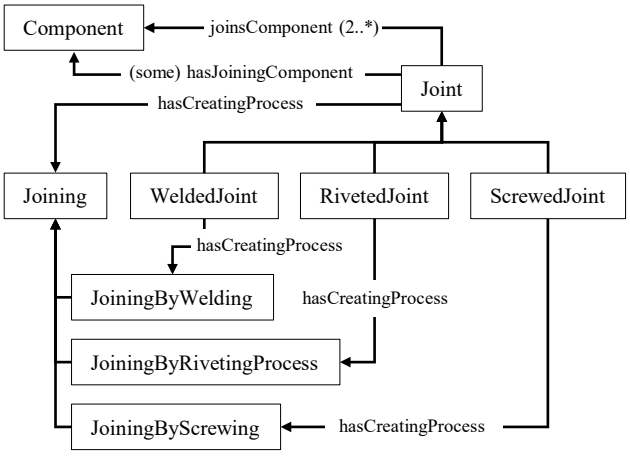

Figure 5: Data model of relationships between Joint, Joining and Component

The property *hasProductDefinition* optionally refers to a product definition in form of a part or an assembly. Sub-assemblies are represented by the fact that an assembly functions as the product definition of a *Component*.

The relationships between components, joints and joining methods are also modelled in the knowledge graph (see Figure 5). Joints are defined by the *Joint* class and its subclasses for *WeldedJoints*, *RivetedJoints* and *BoltedJoints*. The object property *hasCreationProcess* with the domain *Joint* and the range *Joining* is used to link the respective connection types with the corresponding joining manufacturing process.

There are two further properties with the domain *Joint*. The *joinsComponent* property is used to specify which components are joined. At least two elements must be part of a joint. The sub-property *hasJoiningComponent* specifies the joining component. A connection requires a joining element which itself is a connected element.

5. Extraction of Connections

Having introduced the data model of the knowledge graph, we will now explain in the following sections our method to extract connections from STEP files. The method is divided into three main steps: (1) extraction of the assembly hierarchy, (2) extraction of connections and joints and (3) export as knowledge graph.

5.1. Extraction of Assembly Hierarchy

In the first step of our extraction approach, the structure of components and sub-assemblies in the assembly is extracted. The first step is further divided into the following three sub-steps:

Extraction of components: In STEP files, all components of assembly and sub-assemblies are represented by instances of the entity *assembly_component_usage* [8]. All instances of these entities are selected by the algorithm and are processed further in the next step.

Determination of component types: After that, we determine the type of all selected instances. Assemblies are identified by the fact that the attribute *related_product_definition* matches the attribute *relating_product_definition* of another component (see Figure 6). An assembly that appears as a component of another assembly is referred as a sub-assembly.

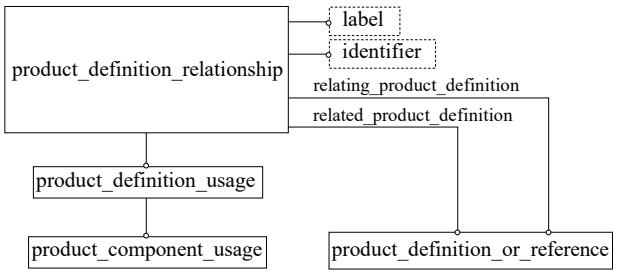

Figure 6: Excerpt of STEP-Schema (component of an assembly) according to [3]

The remaining types of components are identified by their *product name* and *ID*. In the ID and name strings we search for standard designations of screws, nuts, washers or rivets to match those entities to the respective concepts in the knowledge graph. The standard designations are retrieved from the knowledge graph.

In most cases, a component has a product definition in a STEP-file (e.g. a metal sheet or a screw). Weld seams are an exception: They occur as an annotation in the construction plan and are not produced independently of the assembly. To cover that case, we search in the *ID* and name strings for the string "weld" to identify weld seams.

Assign components to (sub-)assemblies: Once the type of each component has been determined, we assign each component to the assembly where the *related_product_definition* matches the *relating_product_definition* attribute of another component.

All components that cannot be assigned to a sub-assembly in this way are assigned to the main assembly

5.2. Extraction of Connections and Joints

In the second step of our extraction approach, connections between components of specific types are identified based on their categorization and hierarchy. The second step is further divided into the following three sub-steps:

Determination of orientation: In the first sub-step we determine the translation and rotation of all components. In STEP, the entity *context_dependent_shape_representation* links between product and the topology of components [8] (see Figure 7.)

For each instance of *context_dependent_shape_representation*, we derive the *product_definition* traversing the attributes *represented_product_relation* and *definition*.

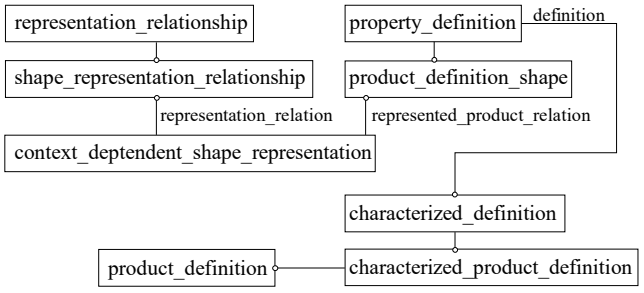

Figure 7: Excerpt of STEP-Schema (form representation relationship) according to [3]

First, the shape representation relationship of the type *shape_representation_relationship* is determined for each component. This is done by extracting all instances of *context_dependent_shape_representation* from the STEP file. Starting from the instances, the *represented_product_relation* attribute can be used to access an instance of the type *product_definition_shape*, which in turn allows access to the product definition. This attribute is matched with the product definitions of the components in order to assign an instance of

the *shape_representation_relation* entity to each component. This is the *representation_relation* attribute of the *context_dependent_shape_representation* type.

The placement of the component can be derived via a shape representation relation. The rotation is represented by the two attributes axis and *ref_direction*. The orientation of the z-axis of the component is determined by the axis, while the *ref_direction* represents the orientation of the x-axis.

Determination of bounding boxes: To simplify the calculation of collisions, the components can be approximated as bounding boxes, which simplifies the shape of the objects and reduces the complexity of the collision detection algorithm at the expense of accuracy [18]. Figure 8 illustrates the bounding boxes of an example assembly in which two metal sheets are bolted together.

The bounding box is calculated on the basis of the corners of the components. The extreme values in the x, y and z directions are determined from the corners found for each component in order to define the boundaries of the shell. The corners of the cuboid, which have been determined in this manner, are then transformed into the global coordinate system using the previously determined translation and rotation.

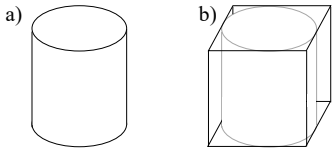

Figure 8: Exemplary bounding boxes of a cylinder; a) cylinder; b) cylinder with bounding box

Identification of connections via collisions: In the subsequent stage, the bounding boxes of the screws, rivets and weld seams are evaluated for potential collisions with the bounding boxes of other components. This is achieved through the utilization of the separating axis theorem.

The recognized collisions with screws, rivets or weld seams are used to determine which components are connected to each other. If the joining component is in the same assembly as the part with which it collides, then there is a connection between the components. However, if a joining component collides with a component in a sub-assembly, then the sub-assembly of the component is considered to be a connected component. An exception to this is the case of washers and nuts, which are considered to be a joining component if they collide with a screw.

Once the extraction is complete, a complete hierarchy of parts and assemblies is established, along with a set of connections defined by their joining component.

5.3. Export to Knowledge Graph

In the third step of our extraction approach, we populate the knowledge graph with the extracted data. We use the data model introduced in Section 4.

First, an individual of the *Assembly* class is created to represent the main assembly. After that we create instances for the components of the main assembly and assigned them to the assembly individual via the property *isComponentOf*. If an instance of a sub-assembly has not been created yet, it will be created too. The components are then exported in the second step. Subsequently, all direct components are created as individuals of the *Component* class and assigned to the main assembly. Components of sub-assemblies are assigned to those via the property *isComponentOf*. The position and rotation are set using the corresponding datatype properties.

Finally, the connections are exported to the ontology. This is achieved by creating an individual of the appropriate class and referencing the joining components with the property *hasJoiningComponent*.

6. Results

In this section, we illustrate the output of our method on the example of an assembly with a bolted flange connection (see Figure 9). The example assembly consists of ten components (two pipes with flange, four bolts and four nuts). The two pipes are joined via the flanges with four screw/nut connections.

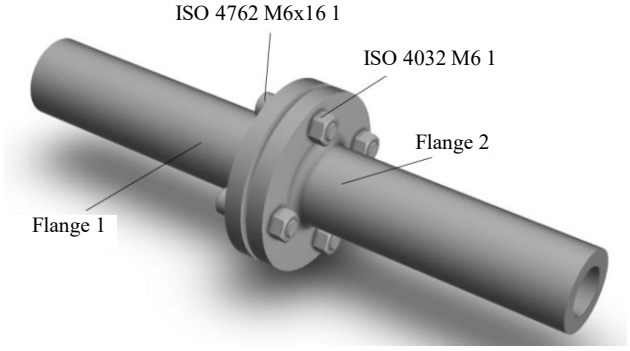

Figure 9: Exemplary flange connection

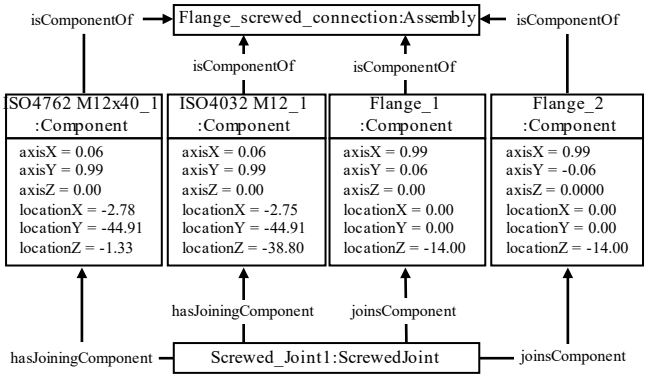

Figure 10: Result of proposed software algorithm

Figure 10 shows a part of a knowledge graph, which describes the example assembly, as an object diagram. The diagram contains the main assembly and four of the ten components with their properties. Further, the object diagram contains one of the four screw connections, comprising the joining components ISO 4762 M12x40_1 and ISO 4032 M12_1, as well as the two connected flanges Flange_1 and Flange_2. The representation of the remaining three screw connections and their respective joining and connected components are omitted in the diagram. Likewise,

the product definition connected to the components using the *hasProductDefinition* property as well as the coordinates and orientation of the component are omitted in the diagram, too.

Although the algorithm works efficiently, it has some accuracy deficits due to the approximation of objects by bounding boxes. The connection detection can show errors with complex components (e.g. with indentations), as the bounding boxes simplify the geometry of a component.

7. Conclusion and Outlook

We presented an automated method to identify connections between components of assemblies from STEP files and to represent them in knowledge graphs. Such a knowledge graph can be used for further analysis of assemblies, e.g. to assess their recyclability or to plan their recycling. Further, this eases the integration of the connection data with other data about the products or production processes. We demonstrated our method with a prototypical implementation of the extraction of bolted, riveted, and welded joints.

For the connection extraction we used collision detection. This was necessary, as major CAD tools do not properly export connection annotations into the STEP format. By improving the STEP exports, the CAD tool developers could contribute to an easier increase in recycling.

In summary, this work is an important step towards developing software systems that evaluate product recyclability to support engineers in designing products for a circular economy. Further, such systems could help recyclers to optimize recycling processes in economic and ecological terms. The use of such systems could contribute significantly to the protection of the global environment.

Author Contribution Statement

Tobias Köhler: Conceptualization, Methodology (Extractor, Knowledge Graph), Writing – Original Draft;

Nico Schmidt: Methodology (Extractor, Knowledge Graph), Software (Extractor, Knowledge Graph);

Jan Martin Keil: Methodology (Knowledge Graph), Writing – Review & Editing;

Diana Peters: Supervision, Writing – Review & Editing

References

[1] M. Al-wswasi, A. Ivanov, H. Makatsoris, A survey on smart automated computer-aided process planning (ACAPP) techniques, Int J Adv Manuf Technol 97 (2018) 809–832. https://doi.org/10.1007/s00170-018-1966-1.

[2] DIN 8589-0:2003-09, Fertigungsverfahren Spanen_- Teil_0: Allgemeines; Einordnung, Unterteilung, Begriffe, Beuth Verlag GmbH, Berlin.

[3] International Organization for Standardization, Industrial automation systems and integration - Product data representation and exchange: Part 1: Overview and fundamental principles 25.040.40, 2021.

[4] M. Kutz (Ed.), Handbook of materials selection, Wiley, New York, 2002.

[5] S. Vajna, C. Weber, K. Zeman, P. Hehenberger, D. Gerhard, S. Wartzack, CAx für Ingenieure: Eine praxisbezogene Einführung, third., vollständig neu bearbeitete Auflage, Springer Vieweg, Berlin, Germany, 2018.

[6] International Organization for Standardization, Industrial automation systems and integration - Product data representation and exchange: Part 203: Configuration controlled 3D design of mechanical parts and assemblies 25.040.40, 2011.

[7] International Organization for Standardization, Industrial automation systems and integration - Product data representation and exchange: Part 214: Core data for automotive mechanical design processes 25.040.40, 2010.

[8] International Organization for Standardization, Industrial automation systems and integration - Product data representation and exchange: Part 242: Application protocol: Managed model-based 3D engineering 25.040.40, 2020.

[9] T.R. Gruber, Towards principles for the design of ontologies used for knowledge sharing, International Journal of Human-Computer Studies 43 (1995) 907–928. https://doi.org/10.1006/ijhc.1995.1081.

[10] w3c.org, OWL 2 Web Ontology Language: Document Overview (Second Edition), 2012. https://www.w3.org/TR/owl-overview/ (accessed 29 October 2020).

[11] G. Schreiber, Y. Raimond, RDF 1.1 Primer: W3C Working Group Note, 2014. https://www.w3.org/TR/rdf11-primer/.

[12] T. Köhler, B. Song, J.P. Bergmann, D. Peters, Geometric Feature Extraction for Additive Manufacturing Parts based on a Knowledge Graph: Preprint, Preprint.

[13] J. Huynh, Separating Axis Theorem for Oriented Bounding Boxes.

[14] S. Gottschalk, M.C. Lin, D. Manocha, OBBTree, in: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 1996, pp. 171–180.

[15] M. Al-wswasi, A. Ivanov, A features subtraction system for rotational parts based on manufacturing and metal removing concepts, Int J Adv Manuf Technol 107 (2020) 1835–1857. https://doi.org/10.1007/s00170-020-05063-w.

[16] Q. Wang, X. Yu, Ontology based automatic feature recognition framework, Computers in Industry 65 (2014) 1041–1052. https://doi.org/10.1016/j.compind.2014.04.004.

[17] S.K. Mohammed, M.H. Arbo, L. Tingelstad, Using semantic Geometric Dimensioning and Tolerancing (GD &T) information from STEP AP242 neutral exchange files for robotic applications, Int J Interact Des Manuf (2023). https://doi.org/10.1007/s12008-023-01242-7.

[18] A. Nischwitz, M. Fischer, P. Haberäcker, G. Socher, Computergrafik, Springer Fachmedien Wiesbaden, Wiesbaden, 2019.