Knowledge Graph of Design Rules for a Context-Aware Cognitive Design Assistant

Armand Huet, Romain Pinquié, Philippe Veron, Frédéric Segonds, Victor Fau

Knowledge Graph of Design Rules for a Context-Aware Cognitive Design Assistant

Armand HUET[1], Romain PINQUIE[2], Philippe VERON[3], Frédéric SEGONDS[1], and Victor FAU[4]

[1] Arts et Métiers Institute of Technology, LCPI, HESAM Université, 75013 Paris, France
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, G-SCOP, Grenoble
[3] Arts et Métiers Institute of Technology, LISPEN, HESAM Université, F-13617 Aix-en-Provence, France
[4] Capgemini DEMS, Toulouse, France

Abstract. [Context] The design of a system shall comply with many design rules that help industrial designers to create high quality design in an efficient way. Nowadays, design rules try to consider all product lifecycle's phases leading to an ever-increasing growth. This context makes the management of design rules a difficult but essential task. This is why many research and industrial works try to automate this task [1, 3, 4]. [Problem] The processing of design rules, which are natural language sentences stored in unstructured documents, requires expert software. Moreover, existing tools interrupt the design workflow and slow down the design process. [Proposition] We propose a Context-Aware Cognitive Design Assistant (CACDA) to support designers who have to satisfy some design rules among "Big Data". First, we describe the CACDA from the user's perspective. Second, we detail the process for modelling unstructured design rules into a computable knowledge graph that will feed the cognitive design assistant. [Future Work] Once our knowledge graph of design rules will be operational, we will concentrate on its processing to retrieve, recommend, and verify design rules. Experiments will also help to determine pros and cons of the design assistant.

Keywords: Design rules, product design, knowledge graph, context awareness, knowledge management, cognitive assistant.

1 Introduction

[Context] According to Clakins et *al*. in [1], design rules synthesize the knowledge of a company and indicate how to create a proven design. They improve product quality, as well as decrease design time and costs. The definition and management of design rules is consequently a crucial design activity.

[Problem] The number of design rules is increasing due to the complexity of modern products (ElMaraghy et *al*. in [2]), legal constraints and DfX expectations. A designer has to search through a large collection of rules to find the ones to satisfy. The mushrooming of design rules makes their retrieval and exploitation laborious, all the more so that they are stored in unstructured documents of hundreds of pages [3].

[Proposal] We propose a knowledge graph of design rules that will feed a Context-Aware Cognitive Design Assistant (CACDA). CACDA is a ubiquitous and intelligent cognitive assistant that uses the information of a design context to facilitate the exploitation (retrieval, recommendation, verification, automation, etc.) of design rules in a CAD environment. On the one hand, we present the services the CACDA provides to the end-users. On the other hand, we detail the knowledge graph that makes design rules computable by the CACDA.

2 Literature review

[Design rules checker] Design rules checking is an active subject in literature with two main tendencies, a procedural and a semantic approach. The procedural approach considers a fix set of design rules and try to ensure that a product respects these rules by detecting all design errors on the model. The rule is represented by a set of algorithms that detect geometric features in the digital mockup that do not respect the rule. The work of Huang et *al*. [4] and industrial tools [5 - 7] illustrate this approach. The main flaws are:

1. Many design rules are natural language statements, that is, an unstructured form of knowledge that is not directly computable. A procedural approach is therefore limited to some design rule types such as geometric constraints.

2. Algorithms representing design rules are complex, which makes their development and maintenance cumbersome.

3. Design rules information is stored in data silos, each representing a specific design context. For example, all rules for milling compliance will be in the silo “design for milling”. Design is a multi-domain process and many design rules do not fit in a single pre-defined context.

[Semantic Network] In [8], Sowa describes Semantic Networks (SN) as “a graph structure for representing knowledge in patterns of interconnected nodes and arcs”. Languages for developing ontology (RDF, RDFS, OWL) and graph-databases (Neo4j,

Grakn, Trinity RDF, Cayley) support the implementation of semantic networks. Such knowledge graphs break silos and focus on the interactions. SN are effective at modeling complex and unstructured knowledge, such as common [9] or specific design information [10]. Several strategies can be used to model design information. For example, MOKA methodology [11] for engineering design knowledge modeling can be used to capture design knowledge [12]. This structured information can then be used by knowledgeware tools. Instead of using a general design knowledge approach like MOKA, we can review specific knowledge modeling strategies for design rule application. Different research teams propose SN models for design rule application [13 – 15]. They use SN to structure CAD model information before performing reasoning on the semantic model in order to detect design errors.

However, this technique requires a lot of work to build a domain specific SN limited to a particular product type. Instead of describing a product type, we plan to use SN to model the information of a design context and create a user-centric cognitive agent that intelligently provides the right design rules to the right designer at the right time.

[Context-Aware Systems] The context of a software user is: "*Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*" [16]. A context aware system uses context information to provide personalized services to the users. As explained by van Engelenburg et *al*. [17]: "*Context-aware systems are systems that have the ability to sense and adapt to the environment*". A context often contains different sub-context [18]. Each sub-context corresponds to a specific source of information that may serve to perform specific recommendation. The author argue that a social context is crucial for information retrieval "*where other people's preferences must be taken into account*". For example, when searching a new product to buy, other people advices can influence our expectations. The data model of a context depends on the application domain and the services it delivers. Dhuieb et *al*. propose a context-aware architecture to present manufacturing knowledge to workers in factories [19]. These technologies need to be adapted to our research problem. Pinquié et *al*. [20] paved the way for a graph-oriented data model of a design context. The data model consists in five sub-contexts: Social, Semantic, Operational IT, Engineering and Traceability contexts. This paper continues this work by presenting the actual implementation of the knowledge graph and the functional architecture of the Context-Aware Cognitive Design Assistant (CACDA) to facilitate the exploitation of design rules.

3 Capture design rule data for an efficient management

This section will present the Context-Aware Cognitive Design Assistant from a user perspective before detailing the underlying knowledge graph that structures the design rules.

3.1 A Context-Aware Cognitive Design Assistant (CACDA)

The Context-Aware Cognitive Design Assistant (CACDA), which is an intelligent cognitive assistant, aims at supporting designers to provide computer-aided design solutions free of errors. Therefore, the CACDA needs to have access to a computable structure of design rules while being context-aware, that is, sensing and reacting based on the design context. So far, our CACDA focuses on four services provided to three stakeholders.

[Knowledge engineer] The knowledge engineer is responsible for the development and maintenance of the knowledge graph that structures the design rules. He performs the basic Create, Read, Update, and Delete operations on nodes and edges. Fig. 1 illustrates that the CACDA shall enable the knowledge engineer to convert the unstructured manual designs into structured engineering and semantic sub-context. A sub-context is a sub-graph of the knowledge graph. The extraction of specific design information from unstructured documents is an entire and active field of research [21 - 23] but is out-of-scope in our research study. Indeed, we assume that the systematic digitalization of knowledge will lead to the disappearance of documents and knowledge engineers will directly enter the design rules in the CACDA. Therefore, the main function of the CACDA is to transform a list of unstructured design rules into a structured computable knowledge graph.

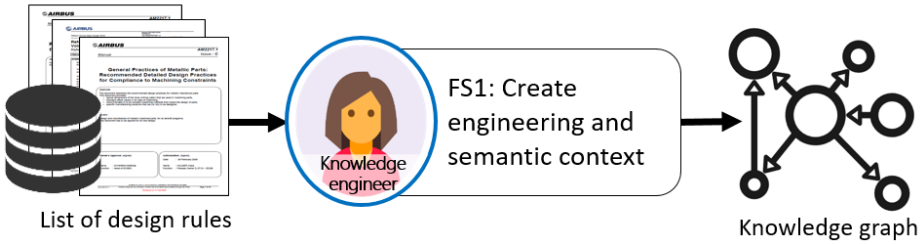

Fig. 1. CACDA service 1: Transform a list of unstructured design rules into a knowledge graph.

[Designer] The CACDA shall enable a designer to retrieve relevant design rules to be satisfied while designing in CAD environment. Path distances and semantic similarities applied to the knowledge graph [24] help to recommend design rules according to the design context. The designer can also query the knowledge graph to search for design rules (full-text search, faceted search, etc.). After their selection, design rules shall appear in the CACDA interface. Fig. 2 and Fig.3 present the service in which the design interact with the cognitive design assistant.

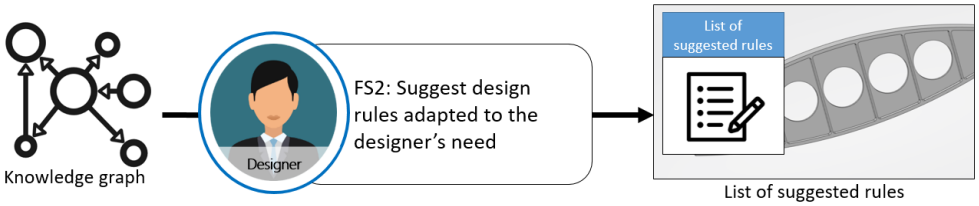

Fig. 2. CACDA service 2: Suggest a design rule list to apply for knowledge graph analysis

Fig. 3. CACDA service 3: Guide the designer in the application of suggested design rules.

[Expert] The expert has a deep understanding of the domain and is consequently able to prescribe new design rules to consider as soon as possible in the design process. Fig. 4 shows that he is the one who suggests new design rules to the CACDA and who appreciates the relevance of the recommendations.

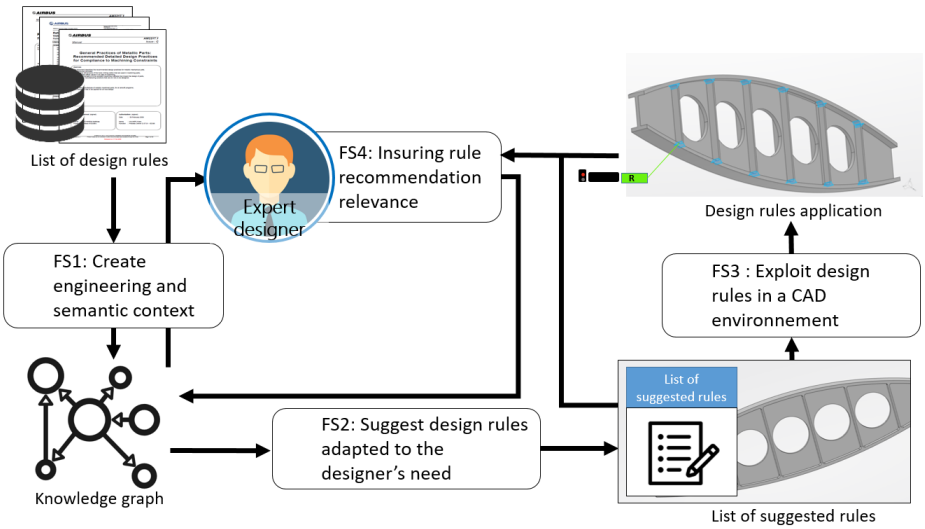

Fig. 4. CACDA service 4: Insure that design rule suggestions are relevant to the designer's context

3.2 Implementation of design rules in the CACDA demonstrator

The previous chapter described the main services the CACDA shall provide to the stakeholders. To provide such support, the CACDA needs design rules structured in a computable knowledge graph. In this section, we present how the processing of design rules and the design context leads to a computable knowledge graph. The CoreNLP toolkit [25] serves for the natural language processing. Ontologies such as ConceptNet [26] and WordNet [27] are key linguistic and common knowledge that enriches the knowledge graph implemented with the NoSQL graph database Neo4J [28]. Python is the pivot language and Dash is the framework that facilitates the prototyping of the web-based user interfaces.

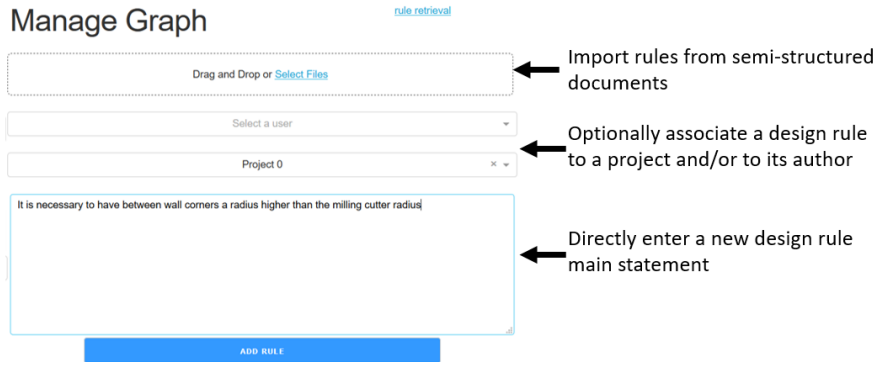

Fig. 5. CACDA knowledge engineer user interface for typing design rule

The knowledge engineer feeds the CACDA with raw design rules and extra knowledge, such as lists of acronyms, glossaries, etc. The semantic processing pipeline of the CACDA structures the knowledge into a computable graph. Fig. 5 illustrates the knowledge engineer user interface from which he can input raw design rules or upload a document containing design rules. For instance, the knowledge engineer types the design rule: "*It is necessary to have between wall corners a radius higher than the milling cutter radius*", extracted from the chapter "*Standard value of wall corner*" of an aircraft design manual. We consider that the CACDA automatically captures the design rule as written in the source document. We will systematically reuse this design rule to illustrate the CACDA.

After entering the design rule, the semantic processing pipeline of the CACDA converts it into a structured graph representation. The graph of design rules forms the so-called semantic sub-context. Figure 6 shows that the semantic sub-context is a sub-graph of the knowledge graph that creates linguistic associations among keywords (nouns, verbs, adjectives, and adverbs). Thus, a full text search of the keyword "*wall*" would return two design rules.

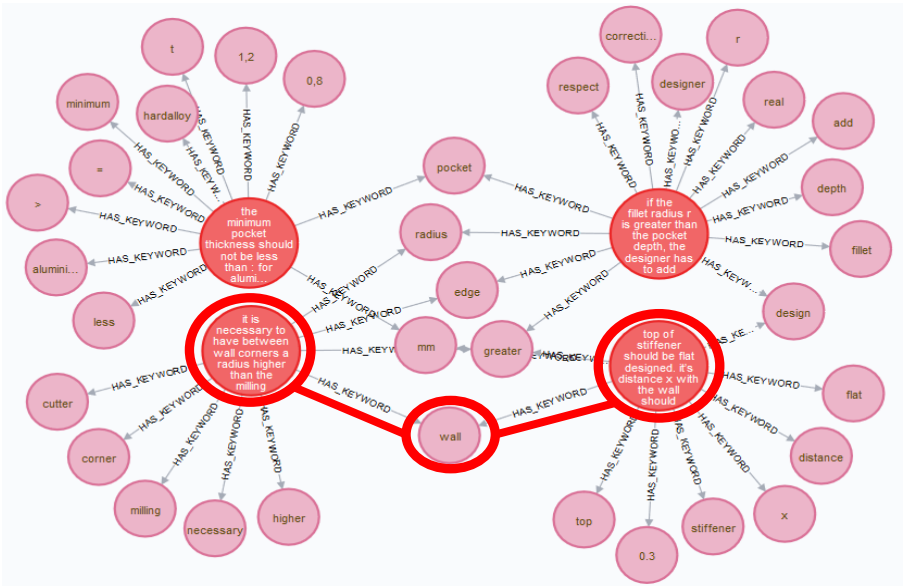

Fig. 6. Graph representation of four design rules (in red) with their keywords (in pink)

Full text search is very limited as it supposes that the designer knows what to look for, whereas he does not really know what rules shall be satisfied. There is therefore a need to expand queries with relevant keywords and allow the navigation in the knowledge graph. State-of-the-art strategies for short statement analysis suggest enriching the data with external semantic resources [29 - 31]. To expand keywords with linguistic features (synonyms, homonyms, meronyms, etc.), we integrate the WordNet thesaurus [32]. Before expanding keywords with relevant terms, we disambiguate each keyword according to the existing semantic sub-context. Figure 7 represents the result of this process on the keyword "*corner*". By considering all the terms linguistically linked to the keyword "*corner"*, the CACDA facilitates the retrieval of unmatched but relevant design rules. Moreover, the richer is the knowledge graph, the more pertinent will be the semantic exploration. For instance, if a designer consults a design rule, the CACDA can recommend similar rules that are very likely to be of interest.

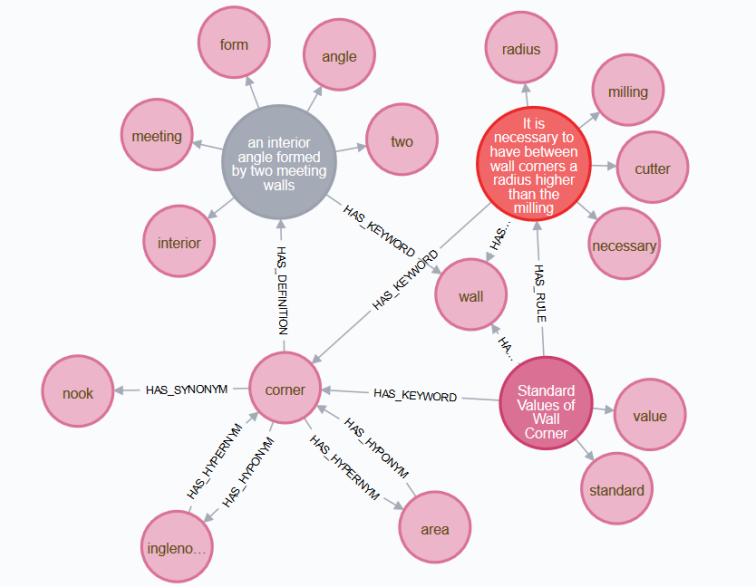

Fig. 7. Extract of a design rule's semantic sub-context with design rule (red), definition (grey), keywords (pink) and chapter (purple)

The semantic sub-context is not enough to retrieve all relevant design rules according to a design context. Indeed, linguistic relationships help to navigate among rules but other aspects are of interest. For instance, as a designer, the CACDA can recommend me design rules because colleagues with a similar profile have satisfied some rules that I oversee. Figure 8 shows how to model the social context in the knowledge graph. By logging designer's activity, we can use collaborative filtering to suggest design rules.

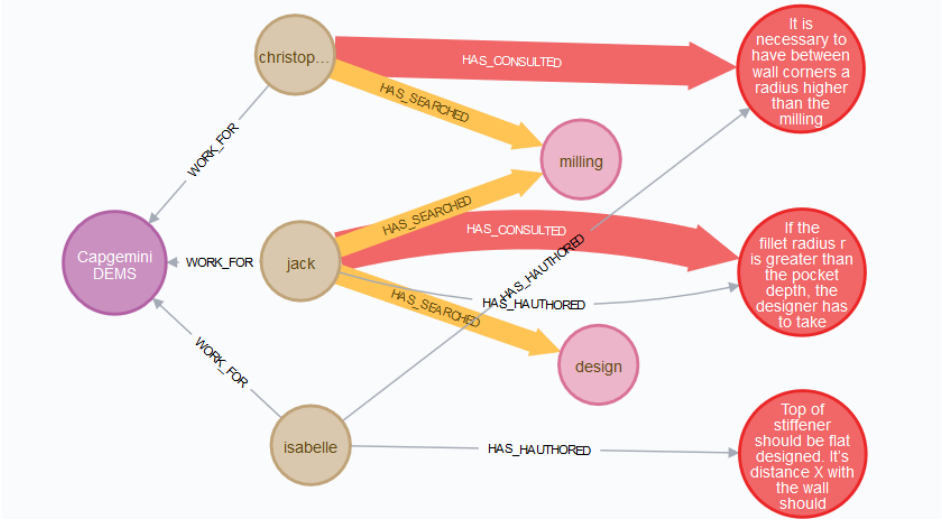

Fig. 8. Design rules with the relationships between the semantic and social sub-contexts. Design rules (red), keywords (pink), users (brown), company (purple)

Social and linguistic relationships are key elements for navigating across design rules, but IT information is also relevant. For instance, if the designer is designing a part using CATIA V5, the name of features in the tree or the name of CAD operations are key information for retrieving design rules. The so-called IT context is also a sub-graph of the knowledge graph that formally defines the current computer-aided design context (software, workbench, operation, etc.). Figure 9 illustrates an instance of an IT context.

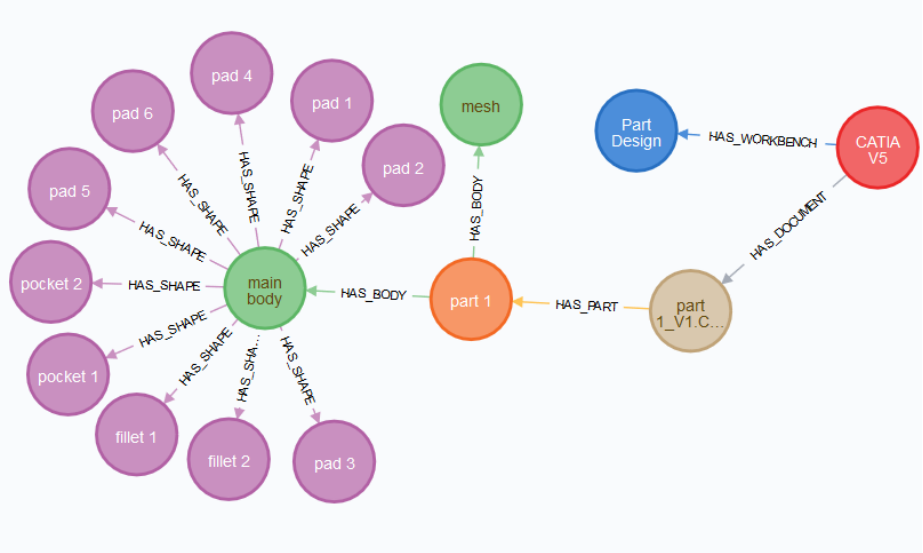

Fig. 9. IT sub-context of a CAD document opened in CATIA V5 with software (red), workbench (blue), document (brown), part (orange), bodies (green) and shapes (purple).

4 Conclusion and future work

The ever-increasing number of design rules, which are multi-domain unstructured knowledge stored in large documents, makes their application laborious. We propose a Context-Aware Cognitive Design Assistant, that is, an intelligent cognitive assistant that facilitates the application of design rules while performing computer-aided design tasks. The assistant relies on a semantic network and design context awareness. In this paper, we describe the services the CACDA provides to the stakeholders and we detail how the processing pipeline structures the design rules and the design context into a computable knowledge graph.

In future work, we will continue the prototyping task, with an emphasis on the analysis of the knowledge graph for recommending and verifying design rules.

References

1. Calkins, D. E., Egging, N., & Scholz, C.: Knowledge-based engineering (KBE) design methodology at the undergraduate and graduate levels. development, 21. (1999).
2. ElMaraghy, W., ElMaraghy, H., Tomiyama, T., & Monostori, L. Complexity in engineering design and manufacturing. CIRP annals, 61(2), 793-814. (2012).
3. Kassner, L., Gröger, C., Mitschang, B., & Westkämper, E.: Product life cycle analytics-next generation data analytics on structured and unstructured data. In: CIRP Conference on Intelligent Computation in Manufacturing Engineering (Vol. 33, pp. 35-40). (2014).
4. Huang, B. et al.: An automatic 3D CAD model errors detection method of aircraft structural part for NC machining. Journal of Computational Design and Engineering, 2(4), 253-260. (2015).
5. Dfmpro. https://dfmpro.geometricglobal.com/
6. Siemens NX Checkmate. https://www.plm.automation.siemens.com/en_us/Images/2504_tcm1023-11882.pdf
7. Dewhurst, B.: DFMA. https://www.dfma.com
8. Sowa, J. F.: Semantic networks. John_Florian_Sowa isi [2012-04-20 16: 51]> Author [2012-04-20 16: 51]. (2012).
9. Wu, F., & Weld, D. S.: Open information extraction using Wikipedia. In Proceedings of the 48th annual meeting of the association for computational linguistics (pp. 118-127). Association for Computational Linguistics. (2010).
10. Cheong, H., Li, W., Cheung, A., Nogueira, A., & Iorio, F.: Automated extraction of function knowledge from text. Journal of Mechanical Design, 139(11). (2017).
11. García, L. E. R., Garcia, A., & Bateman, J. An ontology-based feature recognition and design rule checker for engineering. In Workshop "Ontologies come of Age in the Semantic Web"(OCAS2011) 10 th International Semantic Web Conference Bonn, Germany, October 24, 2011 (p. 48). (2011).
12. Klein, R. Knowledge modeling in design—the MOKA framework. In Artificial Intelligence in Design'00 (pp. 77-102). Springer, Dordrecht. (2000).
13. Skarka, W. Application of MOKA methodology in generative model creation using CATIA. Engineering Applications of Artificial Intelligence, 20(5), 677-690. (2007).
14. Moitra, A., Palla, R., & Rangarajan, A. Automated capture and execution of manufacturability rules using Inductive Logic Programming. In Twenty-Eighth IAAI Conference. (2016).
15. Fortineau, V., Fiorentini, X., Paviot, T., Louis-Sidney, L., & Lamouri, S.: Expressing formal rules within ontology-based models using SWRL: an application to the nuclear industry. International Journal of Product Lifecycle Management, 7(1), 75-93. (2014).
16. Dey, A. K.: Understanding and using context. Personal and ubiquitous computing, 5(1), 4-7. (2001)
17. van Engelenburg, S., Janssen, M., & Klievink, B. Designing context-aware systems: A method for understanding and analysing context in practice. Journal of logical and algebraic methods in programming, 103, 79-104. (2019).
18. Ruthven, I.: Information retrieval in context. Advanced Topics in Information Retrieval. In: Melucci M., Baeza-Yates R. (eds) Advanced Topics in Information Retrieval. The Information Retrieval Series, vol 33. Springer, Berlin, Heidelberg, 187--207 (2011)
19. Dhuieb, M. A., Laroche, F., & Bernard, A.: Context-awareness: a key enabler for ubiquitous access to manufacturing knowledge. Procedia CIRP, 41, 484-489. (2016).
20. Pinquié, R., Véron, P., Segonds, F., & Zynda, T.: A Property Graph Data Model for a Context-Aware Design Assistant. In: IFIP International Conference on Product Lifecycle Management (pp. 181-190). Springer, Cham. (2019).

21. Cheong, H., Li, W., Cheung, A., Nogueira, A., & Iorio, F.: Automated extraction of function knowledge from text. Journal of Mechanical Design, 139(11). (2017).
22. Shi, F., Chen, L., Han, J., & Childs, P.: A data-driven text mining and semantic network analysis for design information retrieval. Journal of Mechanical Design, 139(11). (2017).
23. Pinquié, R., Véron, P., Segonds, F., Croué, N.: Natural language processing of requirements for model-based product design with ENOVIA/CATIA V6. In: Bouras A., Eynard B., Foufou S., Thoben KD. (eds) Product Lifecycle Management in the Era of Internet of Things. PLM 2015. IFIP Advances in Information and Communication Technology, vol 467. Springer, 205--215 (2016).
24. Strobin, L., & Niewiadomski, A.: Recommendations and object discovery in graph databases using path semantic analysis. In: International Conference on Artificial Intelligence and Soft Computing (pp. 793-804). Springer, Cham. (2014).
25. Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations (pp. 55-60). (2014).
26. R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge," no. Singh 2002, pp. 4444–4451. (2016)
27. G. A. Miller, "WordNet: A Lexical Database for English," Commun. ACM, vol. 38, no. 11, pp. 39–41, doi: 10.1145/219717.219748. (1995)
28. Miller, J. J. Graph database applications and concepts with Neo4j. In Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA (Vol. 2324, No. 36). (2013).
29. Abdalgader, K.: Word sense identification improves the measurement of short-text similarity. In: The International Conference on Computing Technology and Information Management (ICCTIM) (p. 233). Society of Digital Information and Wireless Communication. (2014).
30. Shrestha, P.: Corpus-based methods for short text similarity. (2011).
31. Yih, W. T., & Meek, C.: Improving similarity measures for short segments of text. In AAAI (Vol. 7, No. 7, pp. 1489-1494). (2007).
32. Miller, G. A.: WordNet: An electronic lexical database. MIT press. (1998).