Computer-Aided Design 80 (2016) 9–18

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

What are features? An ontology-based review of the literature

Emilio M. Sanfilippo[a,b,*], Stefano Borgo[a]

[a] Laboratory for Applied Ontology (ISTC-CNR), Via alla cascata 56, Trento, Povo, 38123, Italy

[b] Department of Information Engineering and Computer Science, University of Trento, Via Sommarive 9, Trento, Povo, 38123, Italy

ARTICLE INFO

Article history:
Received 9 December 2015
Accepted 2 July 2016

Keywords:
Features
Ontology
Product Lifecycle Management

ABSTRACT

Feature-based product modeling is the leading approach for the integrated representation of engineering product data. On the one side, this approach has stimulated the development of formal models and vocabularies, data standards and computational ontologies. On the other side, the current ways to model features are considered problematic since it lacks a principled and uniform methodology for feature representation.

This paper reviews the state of art of feature-based modeling approaches by concentrating on how features are conceptualized. It points out the drawbacks of current approaches and proposes a high-level ontology-based perspective to harmonize the definition of feature.

1. Introduction

Product Lifecycle Management (PLM) deals with the entire spectrum of data and knowledge concerning the lifespan of industrial products, from the initial phases of requirements elicitation and design, to later phases like manufacturing,[1] selling and product's disposal [1]. In order to be machine-processable and cognitively transparent to software agents and to the variety of stakeholders involved in PLM tasks, product knowledge needs to be specified in languages with formal semantics and driven by experts' conceptualizations about their domains of expertise [2–4].

Computer-based technologies, generically called CAx, used for engineering purposes, like Computer-Aided Design (CAD), Computer-Aided Engineering (CAE), Computer-Aided Manufacturing (CAM), and Computer-Aided Process Planning (CAPP) systems, are traditionally focused on specific modeling tasks. Hence, they are used by different expert communities at different stages of the PLM cycle [5,6]. As a consequence, the conceptual models behind these systems can differ on the type of data they allow to specify and on how such data is organized. For instance, from a CAD perspective, a product is a (possibly complex) geometric entity, while from the CAPP perspective, it is the result of production activities carried out by production tools [7,8].

In order to facilitate PLM tasks, as well as heterogeneous data sharing, multiple-views representation and engineering models interoperability, CAx technologies are asked to be more integrated [9–11]. Additionally, they need to answer the need for embedding qualitative specifications about the engineering intents into quantitative models. Information about what the product is useful for, what are the costs for its production, or the environmental impacts of the manufacturing processes should be coherently available in the models used across the PLM cycle [12–16].

Feature-based modeling approaches have played a relevant role for qualitative knowledge specification and integration since the '70s [17,5,18], and feature-based CAx systems are currently considered the state-of-art technologies for product modeling [6,19]. Much of the research work in this area has been focused on the development of algorithms for the recognition of features in design models, as well as on the development of design-by-features technologies [7,20].

Despite the efforts in the last 40 years, feature-based approaches still have not led to a common understanding of what counts as feature, nor to a principled methodology for feature knowledge specification in formal languages. Each community has proposed its own classification and data model, suitable for their specific tasks, and has built CAx systems on top of such models. As a result, features remain entities differently understood by the different stakeholders. Furthermore, their description is limited to specific modeling concerns and, typically, application-driven.

* Corresponding author at: Laboratory for Applied Ontology (ISTC-CNR), Via alla cascata 56, Trento, Povo, 38123, Italy.
E-mail address: sanfilippo@loa.istc.cnr.it (E.M. Sanfilippo).

[1] We use the term 'manufacturing' in a broad sense to cover various tasks related to product realization like machining and verification.

http://dx.doi.org/10.1016/j.cad.2016.07.001
0010-4485/© 2016 Elsevier Ltd. All rights reserved.

In this scenario, the product models used at different phases can hardly be integrated and shared within and across communities.

In this paper we review the literature concerning the conceptualization of feature notions. Our review is guided by an ontology-based reading of features. For this reason, we will pay particular attention to ontology inspired works which are today widely employed for product data classification and knowledge representation, and are used in advanced CAx systems to foster data sharing and inter-systems interoperability. Differently from previous reviews, which analyze algorithmic procedures for features recognition, feature-based CAx systems or design-by-features methodologies, we focus primarily on the semantics of feature notions, namely on how they are understood across communities, rather than how they are computationally treated in implementation systems. The review is motivated by the lack of progress in understanding the semantics of features. The development and formalization of a broadly applicable knowledge base for PLM requires indeed to systematically analyze the concepts at stake, and that of feature foremost, before moving into application concerns.

2. Research methodology

The initial motivation for this review was the observation that after 40 years of feature-based modeling, there is no unifying view, not to say understanding, of features. If today we ask experts about what a feature is, what properties it has, how we should classify it and, finally, how to model it, we would get puzzled by the variety of answers, by the impossibility to harmonize the answers, even by the lack of a common core for feature understanding. Given the emphasis on feature modeling that we find in the literature and the number of tools based on feature representation that are available, this situation shows that the problem of feature understanding and modeling is deeper than what has been thought so far, and that it does not seem possible to converge to a coherent feature framework by simply exploiting a naïve view of features. In short, we need a feature theory.

This paper aims to look at what we have learned in these years of feature studies: what types of features have been used, what problems have been faced, what is shared among the different views. We consider this is a preliminary step to find a way to move on, perhaps in terms of a change of perspective.

We consider a wide spectrum of publications related to the organization, representation and management of product knowledge and data. The references span from traditional approaches for geometric and parametric feature modeling, to feature taxonomies, object-oriented feature modeling, the development of advanced CAx systems, the application of feature-based approaches to e.g. production costs evaluation, manufacturing verification, machining, functional specification, assembly, among others. A variety of papers dealing with the formal representation of feature knowledge via ontologies is particularly relevant here. Ontologies are today the state of art for transparent modeling, for handling human knowledge in a computer tractable way, for reliably sharing data without loss of relevant information, and for enabling system interoperability. Since there are different types of ontologies, and these can be applied in different ways, we will try to see how successful they have been and what is still missing.

Our analysis of the literature is driven by theoretical insights and formal approaches in ontology engineering [21,22]. In particular, we will point out that the approaches exploited so far in this area of engineering are limited to the so-called semantic technologies, which are quite weak when dealing with sophisticated notions like, we claim, that of feature. Therefore, the state of art presented in Section 3, instead of evaluating technological solutions or algorithmic procedures by which most product models are implemented, focuses on the conceptual models behind these implementations and aims to isolate the foundational assumptions by which features are identified and represented. Accordingly, we investigate how people and organizations understand the domain where they are used and reveal which assumptions drive the development of a certain application [23].

3. State of art: a critical analysis

This section consists of three parts: a report on existing state-of-art reviews concerning feature-based approaches (Section 3.1); an overview of feature definitions focusing on the different understandings of feature notions for PLM purposes (Section 3.2)[2]; and a description of how features are formally represented in information models (Section 3.3).

3.1. Reviews of feature-based approaches

Several survey papers concerning the development and application of feature-based modeling approaches have been presented. The works of Salomon and colleagues [18] and of Bronsvoort and Jansen [24] are amongst the first surveys concerning research issues in feature-based methodologies and technologies. From these publications, it emerges that feature-based models were conceived as being dependent on application constraints and on specific expertise modeling views already at the beginning of the 1990s. As a consequence, engineering models were reusable only at the expense of large re-engineering procedures. To cope with these and other modeling issues, [18,24] proposed to look at methodologies and technologies allowing multiple views integration.

In 1995, van Leeuwen and colleagues [25] discussed the application of feature-based methods for Architecture and Building Information Models and provided a brief review of the state of art of feature-based approaches across mechanical engineering. In this paper they firstly addressed the need of unambiguous and conceptually clear formal models for data modeling and data sharing in engineering; secondly, they addressed the heterogeneous views integration as a main bottleneck for engineering modeling tasks; lastly, they emphasized the necessity to embed qualitative data into quantitative models to allow the formal representation of the designers' intents.

The situation has changed only slightly since then and the described shortcomings can be found in today's modeling methodologies as well. Bronsvoort and colleagues [26] highlight similar shortcomings in existing commercial feature-based systems like the lack of clear semantic specifications for feature notions and the lack of integration for multiple views. According to the authors, features are typically treated in CAx systems as shape macros and, behind morphological aspects, do not support specification of the intents. Amongst the open issues that require further research work, the authors point out the need of new approaches to allow feature models exchange across systems without losing semantic information.

Ma et al. [6] provide an overview of current research issues related to feature modeling, spanning from CAx technologies and methods for feature recognition to data interoperability issues. On the one hand, the authors stress the difficulty of developing shared conceptualizations and formal representations of feature knowledge, given that these tend to be application-dependent; on the other hand, they propose a general layout for feature definition that aims to be application independent (see Section 3.3).

[2] Feature-based approaches in software engineering are behind the purposes of this work.

Table 1
Some feature definitions used in the literature.

Feature definitions
"A region of interest in a part model" [42]
"[...] modeling entities that allow commonly used shapes to be characterised [...] with a set of attributes relevant to an application" [17]
"An information unit describing an aggregation of properties of a product model that are relevant in the scope of a specific view on the product" [43]
"The engineering meaning of the geometry of a part or assembly" [44]
"The characteristics of a product that result from design" [45]
"[...] a physical entity that makes up some physical part" [17]

Several research efforts have focused on the development of algorithms and techniques for automatic feature recognition. The overall goal is to facilitate the integration of CAD systems to downstream CAx applications. Han [27], Han et al. [28], Babic et al. [20] and van den Berg et al. [29] provide reviews of various algorithmic techniques for features recognition in design models, like syntactic pattern recognition, rule-based, graph-based and geometric reasoning methods. The authors emphasize how the strength of recognition approaches is threatened by the different ways the same features can be interpreted depending on the product lifecycle stage or application concerns.

3.2. An overview of existing feature definitions

Features were introduced in the late '70s as modeling elements in CAx systems to represent and reason over both quantitative and qualitative data relevant for product development purposes [17,30–32]. In this sense, feature modeling represents an evolution of computer-based geometric representation, and is nowadays the prevalent approach for product modeling [32,33,11,34]. The early goal was to overcome the repetitive modeling of similar components by developing (software) libraries of pre-defined elements. Initially, features were especially used to organize *part programs* in Computer Numerical Control (CNC) machines [35]. From this perspective, a feature, like a hole or a pocket, is a set of surfaces in a computer model associated to some parameters for manufacturing. Later, it was realized that features could be used to anchor other kinds of qualitative information, namely non-geometric (product) properties useful for modeling tasks. In object-oriented programming, a feature was represented as a class characterized by various attributes; this led to develop taxonomies of features by allowing attribute *inheritance* from the most general to the most specific classes in a taxonomy (e.g. [17,30,36]). For instance, *threaded hole feature* is classified as a subclass of *hole feature* since the former, being characterized via the *threaded* attribute, is more specific than the latter. Soon the initial geometrical view of features was augmented with other types of modeling information making features themselves irreducible to geometrical entities. According to ElMaraghy [37] "[...] features refer to recognizable shapes which cannot be further decomposed, otherwise they will reduce to meaningless geometric entities such as lines, points, and surfaces". Di Stefano and colleagues [38] observed that "the overall aim of feature-based representation is to convert low level geometrical information into high level description in terms of form, functional, manufacturing or assembly features".

On the other hand, from the very beginning some proposed to see features as real-world constituents of products. In their seminal work, Shah and Mäntylä [17] employ two readings of 'feature', on the one side as an "information cluster" for the integrated representation of engineering data, on the other side as "[...] a physical constituent of a part" [17, p. 97]. This ambiguous use of the term has not been without consequences. Brunetti [39], for instance, claims that "a feature is not limited to physical elements and exists only in the world of information models" (see also [40]). From this perspective, a feature is a modeling element in a CAx system used to convey geometric and non-geometric information about the product under design. On the other side, Nepal et al. [41] take features to be "meaningful real world entities to which one can associate construction-specific information". These remarks show that the meaning of 'feature' is not fixed, it is chosen depending on the application and context and, unfortunately, without a systematic treatment of its semantics. Table 1 presents some most recurrent definitions for 'feature'. The double understanding of feature, a modeling element and a physical entity, emerges clearly from the definitions.

As said, historically much work in feature-based product modeling has been focused on *form features*, namely feature specifications in terms of shapes recurrently used for product development purposes like hole, slot, pocket, boss and chamfer [46,36]. These features are called *simple* (or *primitive*, *atomic*) since they are not decomposable in smaller units. In contrast, features like counterbore are called *composite* (or *compound*), because they result from the aggregation of simple features, e.g. two holes [24,7]. However, the exploitation of feature-based modeling to design, manufacturing planning or engineering analysis [11,28,47–49], among others, led to feature models and terminologies driven by application concerns, so that the information attached to features is tuned to a given product lifecycle phase or to a specific application [17,50,18,6]. Table 2 lists some feature categories currently used across the literature. The table is neither an exhaustive list of the variety of categories proposed, nor shows the relationships between these categories. Assembly features, for instance, are sometimes conceived only from a geometric perspective [43,51], while in some other cases their description is enriched with manufacturing details [52–54]. Currently, there is no shared methodology for feature classification, as it depends on application requirements and scenarios; doubts have also been raised about the possibility of an exhaustive categorization of all feature types [28,7].

The dependency on applications has led to the characterization of the very same feature in different ways depending on the modeling perspective one adopts [18,26,40]. A through hole, for instance, is classified as a *functional feature* if described from the perspective of its functionality in a product, while it is classified as a *machining feature* if emphasis is put on the operations required for the hole realization [24]. Interoperability problems arise among different classifications, because the semantics of the represented notions change. A hole as a functional feature, for instance, is meant as a void space in a product, sometimes called a *negative volume* [69]. It is because of a void that an assembly functionality can be attributed to a hole used to accommodate a screw. On the other hand, process planners reason in terms of volumes of material to be removed from the workpiece undergoing a manufacturing process. Thus, a hole as a machining feature is a product's part to be removed from the workpiece.

Nowadays, the notion of feature is used with a variety of meanings for product modeling purposes, not only with reference to holes, bosses and the like, but "anything having an attribute of interest" [70]. Features include components used for assembly, but also qualitative characteristics like colors, dimensions and shapes among others [63,71,58]. This generality is not surprising: from the very beginning some have proposed to understand features as any element in a product that is relevant for product development

Table 2
A partial list of feature categories (ordered alphabetically) with use information and examples.

Feature class	Feature use	Example	Reference
Assembly feature	Used to represent assembly knowledge	Shaft for assembly	[43,51–54]
CAE feature	Used to represent engineering analysis knowledge	Stress analysis feature, fluid flow analysis feature	[47,48,55–57]
Component feature	Used to represent components	Wall, column, screw	[24,58,59]
Form feature	Used to represent elements characterized via shape properties	Hole, pocket, chamfer	[60,18,24,27,28,52,61,62]
Functional feature	Used to represent functional knowledge	Hole for assembly	[18,24,26,63–65]
Geometric feature	Used to represent a geometric element	Surface, edge, vertice	[60,17,24,27]
Machining feature	Used to represent the effects of machining processes	Amount of material swept during a drilling process	[9,10,19,27,28,47,49,66–68]
Material feature	Used to represent material properties	Ceramic feature	[17,62,61,11]

purposes. In De Fazio and colleagues [72], for instance, feature is defined as "any geometric or non-geometric attribute of a discrete part whose presence [is] relevant to the products function [..]". The work recently proposed by Pourtalebi and Horvath [73] represents a step forward in feature modeling, since the authors propose looking at features as *complex properties* of engineering systems. A feature is not just a product's aspect, such as color and dimension, but is rather a property that is structured in (possibly simpler) further properties. This is an interesting approach devoted to the application of a feature-based methodology to the overall design of engineering systems. From the point of view of this review, it is a new challenging step towards a unified view of engineering features.

The interoperability problem raised by heterogeneous and fragmented treatments of features across engineering calls for representational methodologies that can at least clarify the different types of information at stake. Hopefully, we can also organize such information in a way that is coherent with the different modeling perspectives. This would allow reliable data integration while preserving the intended semantics. Yet, at the moment this does not seem possible and each community adopts its own representational approach relying on specific application requirements [11,19,66,74].

3.3. Representing features in information models

Several initiatives focus on the development of feature specifications in terms of data modeling standards, computational ontologies, taxonomies, etc. for disparate applications within the product lifecycle data modeling.

The ISO standard *Automation systems and integration–Product data representation and exchange*, known as STEP (ISO10303) [75], is considered the most relevant effort towards the standardization of product data [76] and the development of engineering environments for data sharing and management across the entire product life-cycle [77]. STEP provides a set of models formalized in the EXPRESS language [78] as well as an application-independent data format to represent and share product data. The AP224 [79] is the STEP application protocol dedicated to feature-based modeling. It specifies recurrent shapes used in manufacturing scenarios. Similarly, the standard ISO 14649 [80] (STEP-NC), resulting from the combination of STEP with the *Data Model for Computerized Numerical Controllers* data structure, classifies various manufacturing features. STEP-NC relates features to the operations and the tools required for their realization in manufacturing environments. Both STEP and STEP-NC are largely used for feature-based product modeling. The reader can find further information in [81–83].

Ma and colleagues [6,34,84,85] propose a general modeling framework for feature-based information management, which is meant to be general enough to cover all application-driven feature categories (see also [11,86]). The so-called 'generic feature' is represented as a UML class that aggregates both quantitative and qualitative constraints. The proposed approach, however, does not provide an explicit characterization of the meaning of feature. This threatens the ability of computer systems to access data semantics as well as to support inter-humans communication. For example, the assumption that the parties involved in data sharing already agree on a common understanding of features constrains the applicability of the approach to single application tasks or to limited groups.

Some research communities have proposed to use computational ontologies for feature-based modeling and data sharing between CAx systems. In computer science ontologies are logical theories, usually encoded in decidable and tractable languages like Description Logics (DL) [87] and the Web Ontology Language (OWL) [88]. Ideally, an ontology is used to explicitly represent a conceptualization of a domain, that is, to constrain the interpretation of a vocabulary to rule out undesired models and to solve ambiguities concerning the semantics of the employed terms [89,23]. Ontologies are currently used for disparate purposes related to information representation and exploitation, like data sharing and interoperability, data storage and retrieval, knowledge representation and reasoning [21].

The *Core Product Model* (CPM) ontology [13], developed at the US National Institute of Standards and Technology (NIST), conceives a product as an aggregation of form, feature and function, where feature is meant as "a subset of the form of an object that has some function assigned to it". The CPM is reused across different research projects. Dartigues and colleagues [8] propose a CPM extension to integrate CAD and CAPP systems by means of the so-called *Feature Ontology*. The latter is formalized in the Unified Modeling Language (UML) [90] and the Knowledge Interchange Format (KIF) [91]. The class 'feature' is here specified as the aggregation of different (technological, economic, etc.) constraints by which application-driven knowledge can be specified to foster CAD/CAPP integration. Additionally, the *Feature Ontology* includes a taxonomy of form features based on STEP, covering e.g. subtraction and protrusion features, among others.

The *Common Design-Feature Ontology* (CDFO) [92,93] is an ontology for CAD systems interoperability. Various feature classes (e.g. hole, counterbore, countersunk) are extracted from systems like Catia V5, Pro/ENGINEERING and SolidWorks, and classified into an OWL taxonomy. For example, a general class for hole features, taken from CATIA, is specialized via the SolidWorks classes 'simple drilled', 'tapered drilled', 'counterbore drilled', etc.

Deshayes and colleagues [94] propose a formal representation in KIF of manufacturing processes by means of the *Process Specification Language* standard (PSL, ISO 18629) [95]. The ontology thus includes notions like tool, workpiece, volume and machine tool, which are relevant for process representation. In this framework, features are the result of cutting operations by machining tools. The terms used in the ontology are however only informally defined in natural language, rather than constrained in a formal manner. Further examples of PSL-based feature representations are in [96].

Researchers at the Wolfson School of Mechanical and Manufacturing Engineering at Loughborough University propose different ontologies to deal with product lifecycle information. The ontologies are formalized in the Extended Common Logic Interchange Format (ECLIF), which is the extended version of Common Logic embedded in the commercial HighFleet environment used for ontology development. As an extension of Common Logic, ECLIF allows the formalization of more expressive constraints than OWL, such as e.g. *n*-ary predicates. Usman and colleagues [97] propose the *Manufacturing Core Ontology* (MCCO) as a common semantic foundation for knowledge representation in manufacturing; the ontology is extended and exploited in [98] for manufacturability analysis and verification. Among its classes, the MCCO includes 'realized part', 'part version', 'manufacturing facility' and 'manufacturing process', which are associated via different relationships. In these works the notion of feature is understood as ''anything having an attribute of interest'' [70]; e.g., a form feature is a feature that has 'form' as attribute of interest, whereas 'production method' is the attribute of interest for production feature. 'Feature', as a class, is subsumed under 'object', which is meant as a physical entity existing in space and time [99–101]. Imran and Young [54] represent assembly constraints in KIF for data sharing in assembly design and planning. In [102] features are used to share product data across design and machining experts within a company.

Kim and colleagues [60] use the notion of assembly feature to represent joining constraints. An assembly feature is understood as a form feature that is functional to assemble different components. The proposed ontology includes general classes like 'product', 'feature' and 'manufacturing [process]', but also more specific information concerning e.g. whether two parts are welded via a butt- or T-joint. The ontology is formalized both in OWL and in the Semantic Web Rule Language (SWRL) [103].

Garcia et al. [71] propose an ontology-based approach for automatic feature recognition to facilitate the linkage between CAD and CAM applications. The authors provide a *CAD Ontology* and a *Feature Ontology*. The former specializes the general class 'CAD features' in 'qualitative features' and 'quantitative features'. Then, 'materials', 'colors' and 'primitives' (e.g. 'line', 'arc') are subsumed under 'qualitative features', whereas 'angle', 'point' and 'parameter' under 'quantitative features'. The *Feature Ontology* is used to model more specific classes, such as 'round slot', 'circular hole', among others. Both ontologies are expressed in OWL and are related to each other by a formal map.

In the same direction, Wang and Yu [104] present an ontology that is split into two modules, the *STEP Box* and the *Feature Box*. The former consists of a partial OWL formalization of the ISO10303-AP203 [105] and includes classes for loop, edge, face and surface, among others, which are the basic building blocks of feature classes. The latter is a feature library that describes features as combinations of the *STEP Box* elements using axioms and SWRL rules. For example, the feature class 'through cut' is represented a set of inner wall faces that are circularly connected.

4. Open problems in today's feature-based modeling approaches

The exploitation of feature-based approaches for the representation of features as classes of entities sharing common attributes, such as UML or OWL classes, has been focused on application concerns leaving aside the semantic clarification of the notions as well as the disambiguation of their meanings. Even the approaches that propose a general framework treat features as aggregations of attributes, without addressing the problem of what a feature is meant to represent. Moreover, as seen in Sections 3.2 and 3.3, some treat features as non-physical elements, thus entities in product models that lack spatial and temporal properties, while for others features are real-world elements in physical products. The two views model important aspects that engineers need to take into account, but their integration in an information system requires careful analysis: to claim, e.g., that a non-physical feature constitutes a physical product would easily lead to logical inconsistencies. The development of computational models in languages with formal semantics (e.g. OWL) does not guarantee *per se* the clarification of the represented notions. Formal semantics is a logical tool by which the interpretations of a language can be controlled and constrained to the desired ones. However, it does not support by itself coherent and transparent knowledge representation [106–108]. To provide a concrete example, the codification of standards in Semantic Web languages (see e.g. [109,110]) surely improves the computational tractability of the developed models; differently from models in EXPRESS, for example, one may automatically reason over a STEP model axiomatized in OWL. However, the mere codification does not necessarily lead to the disambiguation of the represented notions. STEP AP224, for example, treats manufacturing features as materials to be removed from workpieces as well as the results of such removal (see [79, §4.1.5]). This can lead to ambiguities, especially when software agents are in play. For example, a hole manufacturing feature can be both a negative space in a workpiece, and a removed amount of material: a direct codification of the AP224 in OWL would not remove this ambiguity.

The ambiguous formal treatment of features is probably determined by the early successes of the introduction of form features, and the subsequent attempts to directly extend these systems to cover non-morphological information [12,111,11,112, 46]. In particular, current approaches suffer from (at least) the following drawbacks:

- *Lack of a general theory to support the analysis and representation of the employed notions.* To overcome this problem, distinctions between the different entities have been introduced ad hoc leading to scattered and application-driven characterizations. Generally speaking, the class of feature includes qualities (shape, weight, dimension, color, etc.), products' components, amounts of material, things like holes, slots and ribs, among others. In order to develop robust information models, one should identify and distinguish these entities, and provide a framework where they can be related to each other and to the product they refer to. This would allow to state that features which are not independent entities, like a color or a hole, cannot be mentioned in a (product) model unless they are explicitly related to the entity on which they depend.
- *Hidden assumptions in the terms' specification.* This problem is common in specialized domains where general terms are assumed to be implicit limited to the domain of interest and, thus, their meaning is not explicitly stated. Examples are notions like (engineering) function, device and activity. In these cases, there is an implicit assumption that members of the community know how to understand these terms. This assumption particularly applies to standards but ontologies, especially when developed within a specific community, rely on the assumption of a shared understanding of the employed notions. As noted in [102,70], experts within the same research institution, or in different departments within the same company, attribute different meanings to the terms daily used for product development. It is therefore relevant that these meanings are formally captured to ensure automated data interoperability and inter-humans communication.
- *Lack of an ontological characterization of features.* Feature-based models formally describe various constraints on features, especially at the morphological level, but do not model ontological constraints about what features are meant to be. Bidarra and Bronsvoort [113], for example, propose an approach to maintain features geometry throughout the modeling process. They do not introduce constraints to bind features

to other entities: a hole can be inserted in a model without being related to some non-feature entity. Along the same lines, Wang and colleagues [104] axiomatize a variety of geometric constraints on features, but no rule is given to bind instances of feature classes to other entities. Ontological constraints of this kind are not only needed to explicitly characterize assumptions on what features are, but also to verify product models against experts' assumptions, see e.g. [54,98].

- *Lack of an approach that describes understandings of feature notions and allows their comparison, integration and possibly unification.* A coherent description of the variety of understandings of feature notions is necessary to clarify the space of engineering features and to develop a solid comparison towards integration and possibly modeling unification. The approaches developed today tend to reduce features to morphological descriptions. In [70], for instance, a screw hole (functional feature) in a design model is declared equivalent (via a formal mapping) to a web hole (machining feature) in a manufacturing model about the same product. The rationale is that features sharing the same geometry can be identified (see [114] for a similar approach). Although geometric properties seem to provide an anchor to identify or at least combine features, geometry does not provide a solid base to deal with non-quantitative properties. The risk is to miss the design intents behind heterogeneous perspectives. In the example in [70], instead of identifying the features, one should integrate the functional and manufacturing constraints they provide so to preserve all available information.
- *Lack of ontological distinctions in formal models.* Approaches like [115,116] use mereo-topological theories (theories based on parthood and connection relationships) for the representation of assembly features. This is a promising line of research since mereo-topologies are robust knowledge representation theories and their exploitation in design and manufacturing has been advocated for almost twenty years [117,118]. However, the parthood relationship has different properties when applied to objects, processes or space; without a previous ontological distinction of the entities, the formal consequences of these theories can easily be misinterpreted.

From this list of problems emerges the need of a theory for engineering concepts, among which feature, that on the one side represents experts' perspectives and application concerns, and on the other is clear on the domain entities at play. Otherwise said, the theory has to capture the semantics of the employed notions as used in the practise of product development, and has to discriminate between domain entities on the grounds of their ontological properties. Thus, the theory has to include an ontology for feature-based representation that, relying on fundamental distinctions like material vs information, object vs process and property vs role, can make sense of the heterogeneity of data in product models.

In order to fulfill this purpose, we reckon on the theoretical insights and formal methods of ontology engineering [106,119] and, in the next section, give the basic elements for a principled characterization of features. We propose this as the basis of a coherent and unifying feature framework.

5. Basic elements for an ontological modeling of features

The analysis of the literature shows that a feature is understood as an entity that is significant for a product lifecycle task, something that is "meaningful" from a modeling viewpoint (see Table 1). This tells us that 'feature' has to be understood as a technical term: something relevant in or for the product lifecycle. Still, not everything is a feature; the product itself is not a feature, it is rather said to *have* features.

Ontologically, one starts by listing the variety of features used in the domain with the goal to discriminate and organize this collection of examples in relevant types. For this step, it helps to remind that sometimes features are information elements, e.g., the description of a geometrical shape, and in other cases physical entities related to products, e.g. a handle (a physical part of a product). In other cases, features are properties of the product, like its weight, or contextual properties, like its cost; other features are capabilities, like when the piece must hold a certain pressure.

Looking at the different examples, one can posit a basic distinction in terms of what we will call I-feature (information feature) and P-feature (physical feature), that is, the distinction between feature as information entity and feature as physical entity:

I-feature: Feature, in the sense of an information entity, is an element of a product model used to reason about the product under design, its manufacturing, assembly constraints, production costs, etc. The information entities symbolically represented in feature libraries used in CAx systems fall within this feature type. Following this perspective, the intended meaning of I-feature is captured in the literature by definitions like: "An information unit describing an aggregation of properties of a product model that are relevant in the scope of a specific view on the product" [43].

P-feature: Feature, in the sense of a physical entity, is an element related to a physical object, typically the product itself. This type of feature is captured in the literature by definitions like: "The characteristics of a product that result from design" [45], or "[...] a physical constituent of a part" [17]. In this sense, a quality (e.g. color, shape), a component (a handle, a door), an amount of material, or a slot, step, chamfer in the product are P-features.

The two readings are strictly related, as P-features are physical entities whose properties are described by corresponding I-features. From this perspective, to claim that a (physical) feature is an entity with "engineering significance" (e.g. [120,17,121]) means that it is described by an information element (I-feature) in a product model; that is, it is because of a product model that a physical element "acquires" engineering significance for a modeling task.

The two readings have been co-existing for at least 20 years but, unfortunately, the distinction has been often blurred and not treated in a systematic manner. Salomons and colleagues [18], for example, refer to I-features when claiming that a feature is "a carrier of product information that may aid design or communication between design and manufacturing, or between other engineering tasks". Similarly, in their seminal work Shah and Mäntylä [17] see a feature as an "information cluster" for integrated product data representation. However, they muddle the I-feature/P-feature notions when adding that "a feature is a physical constituent of a part" [17, p. 97]. From the ontological perspective, the work of Shah and Mäntylä suffers from a conceptual ambiguity, as the distinction between information and physical features is blurred. But why, one could ask, should we keep these two notions apart? The reason lies in the differences in their ontological characteristics. The very same I-feature can be used in two product models, e.g. the same description of a screwdriver handle can be in several variants, like the one with a slot tip and the one with a Philips tip. Instead, a physical feature is related to a specific physical object and is necessarily located in it: if we consider two screwdrivers, no matter how similar, each has its own handle, thus there are necessarily two handle P-features.

The distinction between information and physical elements for product development purposes is quite common in engineering, and is adopted in well-established ontologies and standards. Thus, our suggestion to systematically enforce the I-feature/P-feature distinction in modeling features is well justified and complies with good engineering practices. For instance, the Industry Foundation Classes (IFC) [122] distinguishes between classes like 'IfcTypeProduct' and 'IfcProduct', which are associated by the (reified) relation 'IfcRelDefinesByType'. Accordingly, the former is used to specify information that characterizes instances of the latter, namely properties of physical objects. For example, Maria's car (a physical object) is a FIAT-500N-1957, because it satisfies the constraint defined by the corresponding FIAT-500N-1957-Type. Then, the latter is a list of properties that some physical object has to satisfy [107].

A similar approach, in this case on temporal entities, is employed in PSL (ISO18629) [95] where the class 'Activity' is distinguished from the class 'Activity Occurrence'. In PSL an activity-occurrence is an event occurring at a certain time and satisfying the constraints established by an activity. The very same activity can constrain several activity-occurrences since it is meant as "[...] a repeatable pattern of behaviour" [95]. In this sense, 'Activity' provides the information related to a certain activity-occurrence like the properties (e.g. ordering constraints) that the occurrence has to satisfy. In the Design Ontology [123] the content of a model is understood as a 'proposition', along with the Suggested Upper Merged Ontology (SUMO) [124], namely an abstract entity that lacks space–time location and is represented via a language. Similarly, in the literature about conceptual modeling, a product as a material object is distinguished from its corresponding product type, which refers to properties that the former has to satisfy [125,126].[3]

Following these approaches, I-features are (typically complex) *properties* used to define product types. On the other hand, P-features are entities in space and time that satisfy I-features. For example, a particular physical slot in a product is a P-feature that corresponds to the realization of an I-feature, meaning that the latter specifies, e.g., the form and the functional properties of the former. In particular, any P-feature has a corresponding I-feature, although the latter might not be explicitly represented in a model. Actually, a P-feature may correspond to several I-features since the physical product may satisfy slightly different models depending on the range of parameters, granularity and tolerance. Vice versa, not all I-features are realized as P-features, e.g. when they are introduced as design alternatives. Borrowing the terminology from IFC, we call *typization* the relationship between P-feature and I-feature: if an I-feature 'typizes' a P-feature, then the latter satisfies the constraints established by the former.

Finally, the distinction between I-feature and P-feature allows us to distinguish between properties that physical entities satisfy because they were produced according to a design model, from properties that they satisfy because of their ontological status. A hole feature in a product model, for example, can be a form I-feature, if characterized by shape constraints, and a functional I-feature if characterized by functional constraints. A physical hole in a product, which has been intentionally realized to satisfy both those shape and functional I-features, is a physical feature with its specific form and functionality. These two properties (form and functionality) are intentionally realized according to the design requirements. Instead, there are properties that are not related to the design, for instance that the (physical) hole is a physical and non-material object in space and time, that it depends on the presence and shape of other material objects, and that other material objects can be allocated in it [127,128]. It is this latter type of properties that we have started to exploit in this section in order to suggest how an ontological view of engineering features can emerge via a systematic analysis of their properties.

[3] Having mentioned different languages, one could introduce here the distinction between information and representational supports (the latter being, e.g., a printed document or a software file where the information is coded). After all, the very same I-feature can be represented in a CAD model, in an engineering drawing and in a spatial logical theory. Nonetheless, since representations are needed only for communication purposes, they can be disregarded in discussing the basic distinctions across engineering features.

6. Discussion and conclusions

This paper provided a critical review of the state of art of feature-based product knowledge representation with emphasis on conceptualization problems and on drawbacks of existing unifying approaches. Our work departs from previous reviews of the literature which typically focus on how features are implemented in CAx systems. Instead, our modeling concerns led us to discuss the concepts behind existing feature models, their understanding and their characterization. As a result of the literature review, we claimed that todays lack of interoperability is due to application-driven representational choices in the information systems for product lifecycle. In particular, we argued that a high-level distinction should be set to separate information features (I-feature) and physical features (P-feature). This basic distinction, already adopted in several engineering standards and methodologies, has been disregarded in the literature on features. The consequence is a conceptual ambiguity that has impeded, in our view, the development of a unifying view. In the last section, we motivated this distinction with ontological arguments. The same arguments, we argue, can be exploited to pursue a new approach for feature classification.

From the literature, a feature should be modeled as an entity intentionally created or selected for a product development purpose. Since features are introduced for application purposes, they are the result of experts' agreements, creativity and modeling choices. In this sense, a feature is an entity *intentionally* introduced in the physical product or in its model, thus made (created) or described (selected) on purpose to satisfy certain requirements. Also, an I-feature can be qualitatively and/or quantitatively characterized, which means that the corresponding properties can be given in terms of value ranges, tolerances, granularity, etc. Finally, no feature can exist by itself. Rather, it needs to be related to some other entity, typically the product (or workpiece) for a P-feature and the product model for an I-feature. Nonetheless, we do not exclude the possibility of having features of features, e.g. the color feature of a part feature, although the pros and cons of this modeling flexibility should be better investigated.

From these observations, we conclude that even feature modeling representations can be improved by exploiting the distinction between properties related to application requirements, and properties related to the domain conceptualization. The latter properties, called 'ontological properties', furnish the elements to separate the domain entities in distinct classes. They enforce e.g. a slot assembly feature firstly to be classified as an object that requires to be bound to a non-feature object, and secondly to have a form that allows the assembly. These constraints are different e.g. from those that (ontologically) characterize a component, which in order to realize its functionality needs to be related to an object with specific characteristics, whereas it can still exist as a product (namely, a designed human-made object) even when detached from any object. On the other hand, the former properties, call them 'engineering properties', play a role in specifying constraints that are useful for application tasks. A slot assembly feature, for example, is associated with topological and geometric descriptions, because this is the information necessary in a CAD system; a component has a label and a numeric identifier (ID), because these are required in a product model to handle relevant PLM information.

In the future, we plan to extend the observations in this paper to develop a notion of engineering feature which is based on engineering practices but also conceptually clear and ontologically sound. From this, we will propose a classification of features. The idea is to rely on an established formal ontology in order to provide a logical characterization of the notion of feature and to make explicit the relationships across different kinds of features across models. The goal is twofold: to validate from the theoretical viewpoint the approach that emerged in this review, and to provide a principled way to clarify and extend existing approaches. If this line of research is successful, it will help to guide the evolution of todays feature representation systems towards a shared understanding of how to understand features and of what characterizes them.

Acknowledgments

This work was partially funded by two projects: GECKO (FdF) funded by MIUR (Miur: FdF-SP1-T2.1) and FourByThree funded by the European Horizon 2020 program: grant 637095. We are grateful to Bob Young, Imre Horváth and the reviewers of the paper for their valuable comments.

References

[1] Saaksvuori A, Immonen A. Product lifecycle management. Springer Science & Business Media; 2008.
[2] Bock C, Zha X, Suh H-W, Lee J-H. Ontological product modeling for collaborative design. Adv Eng Inf 2010;24(4):510–24.
[3] Chandrasegaran SK, Ramani K, Sriram RD, Horváth I, Bernard A, Harik RF, Gao W. The evolution, challenges, and future of knowledge representation in product design systems. Comput Aided Des 2013;45(2):204–28.
[4] Salustri FA. A formal theory for knowledge-based product model representation. In: Knowledge intensive CAD. Springer; 1997. p. 59–78.
[5] Dankwort CW, Weidlich R, Guenther B, Blaurock JE. Engineers' CAx education– it's not only CAD. Comput-Aided Des 2004;36(14):1439–50.
[6] Ma Y-S, Chen G, Thimm G. Paradigm shift: Unified and associative feature-based concurrent and collaborative engineering. J Intell Manuf 2008;19(6): 625–41.
[7] Nasr E, Kamrani A. Computer-based design and manufacturing: An information-based approach. Berlin, (Heidelberg): Springer Verlag; 2007.
[8] Dartigues C, Ghodous P, Gruninger M, Pallez D, Sriram R. CAD/CAPP integration using feature ontology. Concurr Eng 2007;15(2):237–49.
[9] Hou M, Faddis T. Automatic tool path generation of a feature-based CAD/CAPP/CAM integrated system. Int J Comput Integr Manuf 2006;19(4): 350–8.
[10] Miao HK, Sridharan N, Shah JJ. CAD-CAM integration using machining features. Int J Comput Integr Manuf 2002;15(4):296–318.
[11] Uddin MM, Ma Y. A feature-based engineering methodology for cyclic modeling and analysis processes in plastic product development. Comput-Aided Des Appl 2015;1–12.
[12] Decriteau D, Pernot J-P, Daniel M. Towards a declarative modeling approach built on top of a CAD modeler. Comput-Aided Des Appl 2016;1–10.
[13] Fenves SJ, Foufou S, Bock C, Sriram RD. CPM: a core model for product data. J Comput Inf Sci Eng 2008;8(1):1–6.
[14] Patil L, Dutta D, Sriram R. Ontology-based exchange of product data semantics. IEEE Trans Autom Sci Eng 2005;2(3):213–25.
[15] Srinivasan M, Sheng P. Feature-based process planning for environmentally conscious machining - part 1: microplanning. Robot Comput-Integr Manuf 1999;15:257–70.
[16] Gaha R, Benamara A, Yannou B. A feature-based methodology for eco-designing parts on detail phase. In: Design and modeling of mechanical systems. Springer; 2013. p. 645–54.
[17] Shah JJ, Mäntylä M. Parametric and feature-based CAD/CAM. Concepts, techniques, applications. John Wiley and Sons; 1995.
[18] Salomons OW, van Houten FJ, Kals H. Review of research in feature-based design. J Manuf Syst 1993;12(2):113–32.
[19] Hoque A, Halder P, Parvez M, Szecsi T. Integrated manufacturing features and design-for-manufacture guidelines for reducing product cost under CAD/CAM environment. Comput Ind Eng 2013;66(4):988–1003.
[20] Babic B, Nesic N, Miljkovic Z. A review of automated feature recognition with rule-based pattern recognition. Comput Ind 2008;59:321–37.
[21] Staab S, Studer R. Handbook on ontologies. Springer Science & Business Media; 2013.
[22] Guarino N. Formal ontology and information systems. In: Guarino N, editor. Proceedings of the first international conferece on formal ontology in information systems (FOIS). Berlin, (Heidelberg): Springer-Verlag; 1998. p. 3–15.
[23] Borgo S. An ontological approach for reliable data integration in the industrial domain. Comput Ind 2014;65(9):1242–52.
[24] Bronsvoort WF, Jansen FW. Feature modelling and conversion—key concepts to concurrent engineering. Comput Ind 1993;21(1):61–86.
[25] van Leeuwen JP, Wagter H, Oxman RM. A feature based approach to modelling architectural information. In: Fischer L, editor. Modeling of buildings through their life-cycle. Proceedings of the CIB W78 workshop. Stanford University; 1995. p. 467–79.
[26] Bronsvoort W, Bidarra R, Nyirenda P. Developments in feature modelling. Comput-Aided Des Appl 2006;3(5):655–64.
[27] Han J. Survey of feature research, Tech. Rep. IRIS-96-346. USC, (USA): Institute for Robotics and Intelligent Systems; 1996.
[28] Han J, Pratt M, Regli WC. Manufacturing feature recognition from solid models: A status report. IEEE Trans Robot Autom 2000;16(6):782–96.
[29] van den Berg E, Bronsvoort W, Vergeest J. Freeform feature modeling: concepts and prospects. Comput Ind 2002;217–33.
[30] Whitney DE. Mechanical assemblies: Their design, manufacture, and role in product development. Oxford University Press; 2004.
[31] Horvath I, Thernesz V. Morphology-inclusive conceptual modeling with feature objects. In: Y. Shuzi, Z. Ji, L. Cheng-Gang (Eds.), Proc. SPIE 2644, fourth international conference on computer-aided design and computer graphics, 1996, p. 563–72.
[32] Bronsvoort WF, Bidarra R, van der Meiden HA, Tutenel T. The increasing role of semantics in object modeling. Comput-Aided Des Appl 2010;7(3):431–40.
[33] Smit MS, Bronsvoort WF. The difference between two feature models. Comput-Aided Des Appl 2007;4(6):843–51.
[34] Tang S-H, Chen G, Ma Y-S. Fundamental concepts of generic features. In: Semantic modeling and interoperability in product and process engineering. Springer; 2013. p. 89–115.
[35] Shah JJ, Rogers MT. Functional requirements and conceptual design of the feature-based modelling system. Comput-Aided Eng J 1988;5(1):9–15.
[36] Poldermann B, Horvath I. Surface design based on parametrized surface features. In: Proc. int. symposium on tools and methods for concurrent engineering. Budapest: Institute of Machine Design; 1996. p. 432–46.
[37] ElMaraghy H. Intelligent product design and manufacture. In: Pahm D, editor. Artificial intelligence in design. London: Springer Verlag; 1991. p. 147–68.
[38] Di Stefano P, Bianconi F, Di Angelo L. An approach for feature semantics recognition in geometric models. Comput-Aided Des 2004;36(10):993–1009.
[39] Brunetti G. Feature-based virtual engineering. In: Soenen R, Olling GJ, editors. Feature based product life-cycle modelling. Conference on feature modelling in advanced design for the life cycle systems (FEATS), June 12-14, 2001, Valenciennes, France. New York: Springer Science Business Media; 2003. p. 19–39.
[40] Brunetti G, Golob B. A feature-based approach towards an integrated product model including conceptual design information. Comput-Aided Des 2000; 32(14):877–87.
[41] Nepal MP, Staub-French S, Pottinger R, Zhang J. Ontology-based feature modeling for construction information extraction from a building information model. J Comput Civil Eng 2013;27(5):555–69.
[42] Wilson P, Pratt M. A taxonomy of features for solid modeling. In: McLaughlin HW, Encarnacao JL, editors. Geometric modeling for CAD applications. North-Holland; 1988. p. 125–36.
[43] Deneux D. Introduction to assembly features: an illustrated synthesis methodology. J Intell Manuf 1999;10(1):29–39.
[44] Wingård L. Introducing form features in product models: a step towards CAD/CAM with engineering terminology (Ph.D. thesis), Stockholm: Department of Manufacturing Systems, Royal Institute of Technology; 1991.
[45] Groover M. Automation, production systems, and computer-integrated manufacturing. Prentice Hall Press; 2007.
[46] Belaziz M, Bouras A, Brun J-M. Morphological analysis for product design. Comput-Aided Des 2000;32(5):377–88.
[47] Xu X. Integrating advanced computer-aided design, manufacturing, and numerical control: Principles and implementations. Information science reference, New York: Hershey; 2009.
[48] Li L, Ma Y. CAD/CAE associative features for cyclic fluid control effect modeling. Comput-Aided Des Appl 2016;13(2):1–13.
[49] Deja M, Siemiatkowski MS. Feature-based generation of machining process plans for optimised parts manufacture. J Intell Manuf 2013;24(4):831–46.
[50] Shah JJ, Mäntylä M, Nau DS. Advances in feature based manufacturing. Elsevier; 1994.
[51] Ma Y-S, Britton G, Tor S, Jin L. Associative assembly design features: Concept, implementation and application. Int J Adv Manuf Technol 2007;32(5–6): 434–44.
[52] Coma O, Mascle C, Veron P. Geometric and form feature recognition tools applied to a design for assembly methodology. Comput-Aided Des 2003;35: 1193–210.
[53] Van Holland W, Bronsvoort WF. Assembly features in modeling and planning. Robot Comput-Integr Manuf 2000;16(4):277–94.
[54] Imran M, Young RI. The application of common logic based formal ontologies to assembly knowledge sharing. J Intell Manuf 2015;26(1):139–58.
[55] Lee SH. A CAD–CAE integration approach using feature-based multi-resolution and multi-abstraction modelling techniques. Comput-Aided Des 2005;37(9):941–55.
[56] Shephard MS, Beall MW, O'bara RM, Webster BE. Toward simulation-based design. Finite Elem Anal Des 2004;40(12):1575–98.
[57] Hamri O, Léon J-C, Giannini F, Falcidieno B. Software environment for CAD/CAE integration. Adv Eng Softw 2010;41(10):1211–22.

[58] Staub-French S, Fischer M, Kunz J, Ishii K, Paulson B. A feature ontology to support construction cost estimating. AI EDAM: Artif Intell Eng Des Anal Manuf 2003;17(2):133–54.
[59] Pourtalebi S, Horvath I, Opiyo EZ. Fundamentals of a mereo-operandi theory to support transdisciplinary modeling and co-design of cyber-physical systems. In: Proceedings of the ASME 2015 international design engineering technical conferences and computers and information in engineering conference, 2015, p. 1–12.
[60] Kim K-Y, Manley D, Yang H. Ontology-based assembly design and information sharing for collaborative product development. Comput-Aided Des 2006;38:1233–50.
[61] Sy M, Mascle C. Product design analysis based on life cycle features. J Eng Des 2011;22(6):387–406.
[62] Qian X, Dutta D. Feature-based design for heterogeneous objects. Comput-Aided Des 2004;36(12):1263–78.
[63] Brown DC. Functional, behavioral and structural features. In: ASME 2003 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers; 2003. p. 895–900.
[64] Schulte M, Weber C, Stark R. Functional features for design in mechanical engineering. Comput Ind 1993;23(1):15–24.
[65] Umeda Y, Ishii M, Yoshioka M, Shimomura Y, Tomiyama T. Supporting conceptual design based on the function-behavior-state modeler. Artif Intell Eng Des Anal Manuf 1996;10(04):275–88.
[66] Fu M, Ong S-K, Lu WF, Lee I, Nee AY. An approach to identify design and manufacturing features from a data exchanged part model. Comput-Aided Des 2003;35(11):979–93.
[67] Zheng Y, Taib JM, Tap MM. Decomposition of interacting machining features based on the reasoning on the design features. Int J Adv Manuf Technol 2012; 58(1–4):359–77.
[68] Liu J, Liu X, Cheng Y, Ni Z. An approach to mapping machining feature to manufacturing feature volume based on geometric reasoning for process planning. Proc Inst Mech Eng B 2015.
[69] Subramani S, Gurumoorthy B. Maintaining associativity between form feature models. Comput-Aided Des 2005;37(13):1319–34.
[70] Usman Z, Young RI, Chungoora N, Palmer C, Case K, Harding J. Towards a formal manufacturing reference ontology. Int J Prod Res 2013;51(22): 6553–72.
[71] García LER, 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), 10th international semantic web conference, Bonn, Germany, 2011, p. 48–58.
[72] De Fazio TL, Edsall A, Gustavson R, Hernandez J, Hutchins P, Leung H, Luby S, Metzinger R, Nevins J, Tung K, et al. A prototype of feature-based design for assembly. J Mech Des 1993;115(4):723–34.
[73] Pourtalebi S, Horváth I. Towards a methodology of system manifestation features-based pre-embodiment design. J Eng Des 2016;1–37.
[74] Gupta RK, Gurumoorthy B. A feature-based framework for semantic interoperability of product models. J Mech Eng 2008;54(6):446–57.
[75] ISO, industrial automation systems and integration - product data representation and exchange. Part 1: Overview and fundamental principles, 1994.
[76] Szykman S, Fenves S, Keirouz W, Shooter S. A foundation for interoperability in next-generation product development systems. Comput-Aided Des 2001; 33:545–59.
[77] Zhao Y, Habeeb S, Xu X. Research into integrated design and manufacturing based on step. Int J Adv Manuf Technol 2009;44:606–24.
[78] Schenck D, Wilson P. Information modeling: the EXPRESS way. Oxford University Press; 1994.
[79] ISO, Industrial automation systems and integration - product data representation and exchange - Part 224, mechanical product definition for process planning using machining features, 2000.
[80] ISO, Industrial automation systems and integration - physical device control - data model for computerized numerical controllers - Part 10: General process data, iso 14649:2004 edition.
[81] Amaitik SM, Kiliç SE. Step-based feature modeller for computer-aided process planning. Int J Prod Res 2005;43(15):3087–101.
[82] Amaitik SM, Kiliç SE. An intelligent process planning system for prismatic parts using step features. Int J Adv Manuf Technol 2007;31(9–10):978–93.
[83] Bhandarkar M, Nagi R. Step-based feature extraction from step geometry for agile manufacturing. Comput Ind 2000;41:3–24.
[84] Chen G, Ma Y-S, Thimm G, Tang S-H. Unified feature modeling scheme for the integration of CAD and CAx. Comput-Aided Des Appl 2004;1(1–4):595–601.
[85] Chen G, Ma Y, Thimm G, Tang S. Associations in a unified feature modeling scheme. J Comput Inf Sci Eng 2006;6(2):114–26.
[86] Xie Y, Ma Y. Design of a multi-disciplinary and feature-based collaborative environment for chemical process projects. Expert Syst Appl 2015;42(8): 4149–66.
[87] Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider PF, editors. The description logic handbook: theory, implementation, and applications. Cambridge University Press; 2003.
[88] McGuinness DL, Van Harmelen F. et al. OWL: Web ontology language overview [online] 2004 [cited April 20th, 2015].
[89] Guarino N, Oberle D, Staab S. What is an ontology? In: Handbook on ontologies. Springer; 2009. p. 1–17.
[90] Rumbaugh J, Jacobson I, Booch G. Unified modeling language. The reference manual. Pearson Higher Education; 2004.
[91] Genesereth MR, Fikes RE, et al. Knowledge interchange format-version 3.0: reference manual, Tech. Rep. Stanford, (California), (USA): Computer Science Department, Stanford University; 1992.
[92] Abdul-Ghafour S, Ghodous P, Shariat B, Perna E. A common design-features ontology for product data semantics interoperability. In: Proceedings of the IEEE/WIC/ACM international conference on web intelligence. IEEE Computer Society; 2007. p. 443–6.
[93] Abdul-Ghafour S, Ghodous P, Shariat B, Perna E, Khosrowshahi F. Semantic interoperability of knowledge in feature-based CAD models. Comput-Aided Des 2014;56:45–57.
[94] Deshayes L, Beqqali OE, Bouras A. The use of process specification language for cutting processes. Int J Prod Dev 2005;2(3):236–53.
[95] Grüninger M. Using the PSL ontology. In: Handbook on ontologies. Springer; 2009. p. 423–43.
[96] Young RI, Gunendran G, Cutting-Decelle A-F, Gruninger M. Manufacturing knowledge sharing in PLM: A progression towards the use of heavy weight ontologies. Int J Prod Res 2007;45(7):1505–19.
[97] Usman Z, Young R, Chungoora N, Palmer C, Case K, Harding J. A manufacturing core concepts ontology for product lifecycle interoperability. In: Sindered M, Johnson P, editors. Enterprise interoperability. Proceedings for the third international IFIP working conference, IWEI 2011, Stockholm, Sweden, March 23-24, 2011. LNBIP, vol. 76. 2011. p. 5–18.
[98] Anjum N, Harding JA, Young RI, Case K. Manufacturability verification through feature-based ontological product models. Proc Inst Mech Eng B 2012; 0954405412437125.
[99] Anjum N. Verification of knowledge shared across design and manufacture using a foundation ontology (Ph.D. thesis), Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University; 2011.
[100] Imran M. Towards an Assembly reference ontology for assembly knowledge sharing (Ph.D. thesis) Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University; 2013.
[101] Usman Z. A manufacturing core concepts ontology to support knowledge sharing (Ph.D. thesis) Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University; 2012.
[102] Urwin EN, Young RIM. The reuse of machining knowledge to improve designer awareness through the configuration of knowledge libraries in plm. Int J Prod Res 2014;52(2):595–615.
[103] Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M. et al. Swrl: A semantic web rule language combining owl and ruleml [online], 2004. [cited April 20th, 2016].
[104] Wang Q, Yu X. Ontology-based automatic feature recognition framework. Comput Ind 2014;65(7):1041–52.
[105] ISO, Industrial automation systems and integration - product data representation and exchange - Part 203: Application protocol, configuration controlled design 3D designs of mechanical parts and assemblies, 2004.
[106] Guarino N. The ontological level. In: Casati R, Smith B, White G, editors. Philosophy and the cognitive sciences. Hölder-Pichler Tempsky; 1994. p. 52–67.
[107] Borgo S, Sanfilippo EM, Šojić A, Terkaj W. Ontological analysis and engineering standards: An initial study of IFC. In: Ebrahimipour V, Yacout S, editors. Ontology modeling in physical asset integrity management. Springer; 2015. p. 17–43.
[108] Sanfilippo EM, Borgo S, Masolo C. Events and activities: Is there any ontology behind BPMN? In: Garbacz P, Kutz O, editors. Frontiers in artificial intelligence and applications. Proceedings of the 8th international conference on formal ontology in information systems, Vol. 267. Amsterdam: IOS Press; 2014. p. 147–56.
[109] Krima S, Barbau R, Fiorentini X, Sudarsan R, Sriram RD. Ontostep: OWL-DL ontology for step, National Institue of Standards and Technology, NISTIR 7561.
[110] Beetz J, Van Leeuwen J, De Vries B. IfcOWL: A case of transforming EXPRESS schemas into ontologies. Artif Intell Eng Des Anal Manuf 2009;23(01): 89–101.
[111] Cheng Z, Tang S, Chen G, Ma Y. Unified feature paradigm. Semant Model Interoper Prod Process Eng: Technol Eng Inform 2013;117.
[112] Chen G, Ma Y, Thimm G, Tang S-H. Knowledge-based reasoning in a unified feature modeling scheme. Comput-Aided Des Appl 2005;2(1–4):173–82.
[113] Bidarra R, Bronsvoort WF. Semantic feature modelling. Comput-Aided Des 2000;32(3):201–25.
[114] Brunetti G, Grimm S. Feature ontologies for the explicit representation of shape semantics. Int J Comput Appl Technol 2005;23(2):192–202.
[115] Kim K-Y, Yang H, Kim D-W. Mereotopological assembly joint information representation for collaborative product design. Robot Comput-Integr Manuf 2008;24(6):744–54.
[116] Demoly F, Matsokis A, Kiritsis D. A mereotopological product relationship description approach for assembly oriented design. Robot Comput-Integr Manuf 2012;28(6):681–93.
[117] Guarino N, Borgo S, Masolo C. Logical modelling of product knowledge: Towards a well-founded semantics for STEP. In: Proceedings of European conference on product data technology, Citeseer, 1997, p. 183–90.
[118] Sallustri F. Mereotopology for product modelling. a new framework for product modelling based on logic. J Des Res 2002;2:1–15.

[119] Guarino N, Welty CA. Towards a methodology for ontology-based model engineering. In: Proceedings of the ECOOP–2000 workshop on model engineering. LNAI, vol. 1937. Springer; 2000. p. 97–112.

[120] Geelink R, Salomons OW, Van Slooten F, Van Houten F, Kals HJ. Unified feature definition for feature based design and feature based manufacturing. Comput Eng 1995;517–34.

[121] Vandenbrande JH, Requicha AA. Spatial reasoning for the automatic recognition of machinable features in solid models. IEEE Trans Pattern Anal Mach Intell 1993;15(12):1269–85.

[122] buildingSmart International Ltd. Industry foundation classes (IFC), version 4 - addendum 1 [online, cited April 20th, 2016].

[123] Štorga M, Andreasen MM, Marjanovic D. The design ontology: Foundation for the design knowledge exchange and management. J Eng Des 2010;21(4): 427–54.

[124] Sowa JF. Knowledge representation: logical, philosophical, and computational foundations. Course Technology; 1999.

[125] Dahchour M, Pirotte A, Zimányi E. Materialization and its metaclass implementation. IEEE Trans Knowl Data Eng 2002;14(5):1078–94.

[126] Pirotte A, Zimányi E, Massart D, Yakusheva T. Materialization: a powerful and ubiquitous abstraction pattern. In: Proceedings of the 20th international conference on very large data bases. Morgan Kaufmann Publishers Inc.; 1994. p. 630–41.

[127] Galton A. Qualitative spatial change. Oxford University Press; 2000.

[128] Hahmann T, Brodaric B. The void in hydro ontology. In: Donnelly MB, Guizzardi G, editors. Frontiers in artificial intelligence and applications. Proceedings of the 7th international conference on formal ontology in information systems, Vol. 239. Amsterdam: IOS Press; 2012. p. 45–58.