Negative Knowledge and a Novel Approach to Support MCAD Education

Ferruccio Mandorli[1] and Harald E. Otto[2]

[1]Polytechnic University of Marche, f.mandorli@univpm.it
[2]Polytechnic University of Marche, h.e.otto@univpm.it

ABSTRACT

Traditional computer-aided design (CAD) education in mechanical engineering still remains a major challenge today both in industrial settings and in academia. As in many other CAD-related engineering disciplines, there are several shortcomings to be surmounted in the dissemination and development of procedural knowledge and skills in the form of know-how related to the operation of CAD systems. Unfortunately, current educational philosophy does not offer a pedagogy providing sufficient strategic knowledge and understanding to enable students to use CAD systems as intended – that is as knowledge-intensive design and communication tools to properly develop and convey design intent. However, apart from knowing what to do, there is another important aspect to strategic knowledge which is frequently over-looked and ignored in research today, and that is knowing how to avoid serious mistakes. This is a central quality of professional expertise, which is commonly referred to in the literature as negative knowledge. Research presented and discussed in this paper is aimed at providing a framework for negative knowledge and domain knowledge related model evaluation concepts that allow for direct translation of this approach into practice, with the goal of improving learning behavior, skill acquisition and competency building for CAD education in mechanical engineering.

Keywords: CAD education, design intent, feature-based design, negative expertise.
DOI: 10.3722/cadaps.2013.1007-1020

1 INTRODUCTION

An essential part of today's engineering student qualifications, in addition to domain knowledge and problem-solving abilities, is the knowledge and skills to use computer-aided design (CAD) systems in modeling, documentation, and communication of product and service designs within global digital engineering environments, where various computer-aided engineering (CAE) tools and product life-cycle management (PLM) systems are about to become as ubiquitous as CAD systems and tools themselves. In mechanical engineering, over the past decade, feature-based parametric CAD systems have become the de facto standard in professional environments, due to their potential to greatly enhance efficiency of human-driven product design and development processes. However, to translate their potential into actual benefits within a professional environment, an essential precondition is the creation of models that can be easily understood and altered by domain experts (see also discussions in [13, 26]).

Like in many other engineering disciplines, traditional curricula in mechanical engineering for CAD (MCAD) education were mainly concerned with the dissemination and development of procedural knowledge and skills in the form of knowing that or knowing what related to the operation of CAD systems (cf. [16, 26]). Unfortunately, such an educational philosophy does not provide the pedagogy necessary to instill sufficient strategic knowledge (know-how of design/modeling strategies and how to choose between them) and understanding to enable students to use CAD systems as more knowledge-intensive design and communication tools which can properly develop and convey design intent (cf. [4, 5]). This is an inauspicious situation, which is reflected in academia by insufficient MCAD education programs and in industrial settings by a vast number of CAD models that are difficult to understand and to alter efficiently.

However, in the opinion of the authors there is still considerable room for improvement in MCAD education. Such improvement could be introduced immediately if one reconsiders directions and possibilities that have been overlooked for a long time. One of the improvements is concerned with a central quality of professional expertise, namely knowing how to avoid serious mistakes, commonly referred to as negative knowledge. It is also a part of strategic knowledge, complementing the knowledge of what one should do, i.e. being able to avoid choosing a method considered inappropriate in a particular situation with a given goal. Research presented and discussed in this paper is aimed at providing a novel framework for negative knowledge and MCAD specific concepts that allow for direct translation of this innovative approach into educational practice in both industrial settings and academia.

The remainder of the paper is organized as follows. In section 2 some background is given together with an outline of scope and objectives. In section 3 negative knowledge and expertise are discussed along with the introduction of a new framework and its central concepts. This is followed by application examples related to the context of MCAD education in section 4. A summary with conclusions and the outlook for future work is given in section 5.

2 BACKGROUND, SCOPE AND AIMS

2.1 Feature Technology and Design Intent

Initial work on features can be dated back to the early 1980s, when Kyprianou [15] introduced the first form feature taxonomy and also defined basic principles, which then were used to develop automatic form feature recognition algorithms for polyhedric geometric models. At that time, the focus of research was directed towards automatic identification of the procedures which facilitate extraction of information from geometric models, required to support part coding and computer-aided process planning. From the theoretical point of view, as the fields of application of feature technology expanded (from part coding and computer-aided process planning (CAPP) to manufacturing and assembly) it became evident that feature definition and feature taxonomy were highly context dependent; a circumstance that initiated work on feature translation/transmutation, the aim of which was to look at the geometric model from several semantically different points of view.

At that time, although several definitions of features were given, the common underlying concept was that a feature is a set of geometric entities having particular properties, which can be related to a functional and/or technological meaning. In other words, a feature is a way of looking at the shape of a mechanical part from a higher level of abstraction. In engineering design (mechanical design in particular), the relationship between function and shape is a well-known design principle. Hence, the development of an innovative modeling approach based on parametric features, feature-based modeling or design by feature, represents a natural step in the evolution of research in this field. The design by feature paradigm assumes that a mechanical part can be described by a set of features that captures and represents the design intent. An exhaustive review of the literature on feature technology is beyond the scope of this paper. However, a well-documented overview on the work done in this field from the early 1980s to the mid 1990s can be found in [27]. An overview on work done in the past one and a half decades on feature technology, taking into account a stronger linkage of shape, functional meaning, and additional technological aspects such as manufacturing and assembling across phases of the product life cycle by introducing knowledge-based design features and associative features, can be found in [6, 12, 18, 22].

By the mid 1990s, design by feature and parametric modeling had matured to the point that research results were incorporated into commercial MCAD systems, namely SolidWorks, Solid Edge and Pro/Engineer (Creo Elements/Pro). This new generation of 3D modeling systems introduced an innovative and more efficient approach to 3D solid modeling, based on an easier and more intuitive user interaction, though not without caveats. Nowadays, more than one and a half decades since their introduction, the ability of parametric feature-based CAD systems to capture and represent design intent, remains quite limited, at least unless the user adopts specific modeling strategies. This is a critical issue, which is reflected – among other places – in the non-trivial problem of consistent model alteration during the product development process.

A key aspect of a complete and thorough implementation of the feature-based paradigm within a CAD system, is the system's capability to manage a dual representation of the part in a sound way. It must be able to produce a feature-based representation, that is a function-based high level of abstraction description of the part, and a boundary representation, that is a shape-based low level of abstraction description of the part geometry. Of course, the two representations must have a one-to-one semantic correspondence, i.e., they have to express the same functional meaning. For example, if there is a hole feature in the feature-based description of the part, the same hole feature must be present as geometry/topology in the boundary representation of the part. This issue was addressed in the late 1990s by several researchers, who introduced the concept of self-validating features and semantic feature maintenance [1, 17]. The so-called *functional molded part* (FMP) module of Dassault Systèmes' CAD system CATIA V5 partially addresses this problem in the context of functional modeling for molded parts and mold tooling. Otherwise, however, none of the commercially available CAD systems is capable of automatically managing a feature-based representation in a sound way by automatically maintaining the semantic correspondence between the feature-based and the geometry-based representation of the internal model. In particular, what is missing is the control the system should have of the feature behavior throughout the whole modeling process.

2.2 Advanced CAD Technology and Current Modeling Practice

Most of today's commercially available feature-based MCAD systems are based on the so-called *feature history*, which is just the chronologically ordered list of modeling commands that are employed by the user to generate the shape aspect of the CAD model. This implementation of design features is quite trivial in nature, thus remaining far below the functional and semantic (in a technological meaning) level of the original feature concept, and providing only a very limited means of system support to capture the design intent. A direct consequence of such an unsatisfactory situation is that CAD users, especially novices, can easily create feature-based models comprised of very complicated and sometimes ill-defined sequences of features. Such CAD models, in most cases, tend to be unnecessarily complicated and sometimes even impossible to alter, because they are "unreadable" in their feature representation. Those models also exhibit a total lack of any logical connection between the shape generated and the feature representation.

Unfortunately, it is quite common in CAD practice and in industrial application for preference to be given to re-modeling from scratch instead of trying to alter an "unreadable" model. It is a somewhat paradoxical situation that we are now encountering advanced CAD technology, which, instead of facilitating easy and efficient model alteration, rather seems to support the creation of such unalterable models. Currently the way CAD vendors are dealing with this problem can be described as a kind of unsophisticated "brutal-force" approach, where new modeling commands are added to existing CAD systems, enabling a direct modification of the model shape by means of stretching, turning and twisting the geometric entities of the boundary model representation. This approach, in the context of commercially available CAD systems, also referred to as *direct modeling* or *explicit modeling*, might be capable of solving some specific problems, especially those related to the alteration of imported models that have lost the feature history structure. However, from a methodological point of view, this is just a postponement of the problem of model reuse due to interpretation/alteration issues.

2.3 Research Objectives and New Directions in CAD Education and Pedagogy

In both academia and industry, didactic pedagogy is still the dominant and most common method of teaching CAD. This represents a traditional, behaviorism-oriented approach with the aim of providing students with basic knowledge and skills. It is considered sufficient for building CAD models with specific CAD systems representing the shape of a part subject to design. Here, the content of the subject matter as related to the modeling process is broken down into individual behavioral steps reflecting algorithms needed to build the topology and geometry of the model and the sequences of commands to operate the CAD system accordingly. Therefore, it is supporting the deficiencies of modern CAD systems, which are heavily based on geometric modeling techniques. This is due to historical reasons related to the development of the design and manufacturing processes that evolved around the geometric shapes of parts and products. In such a scenario of traditional CAD education, learning outcomes obviously lack the components that link different aspects of the CAD model created to actual design intent and the resulting model structure.

Recent work has been aimed at creating awareness of and addressing the most prominent shortcomings and failures of current CAD education. Such efforts have provided new insights and recommendations, although the work is still limited and the results sometimes contradictory. However, it is gradually increasing the empirical body of evidence for improvement, and moving steadily in the right direction. The need for educational exercises in the CAD laboratory, providing opportunities for students to experience both creation of their own models and alteration of models created by someone else, is investigated and discussed in [8, 13, 25]. Work on promoting good design practice by relating model attributes to design intent can be found in [26]. There is a demand for a change of focus in traditional CAD education from the declarative knowledge relating to geometric algorithms and commands required for operating a CAD system, in the literature referred to as *command knowledge*, toward knowledge and expertise which can transcend a particular CAD system. This is discussed, for example, in [2, 5, 28]. This work highlights the need for higher level thinking relating to what is commonly known as *strategic knowledge*, i.e. a knowledge of the different methods of achieving a specific task (goal) and knowing how to choose among those methods. Note that in this context design intent can be considered as falling under the category of strategic knowledge (cf. [11, 13, 25]).

Essential qualities of experts compared to novices, which also apply to skills and expertise in the domain of CAD, are commonly attributed to characteristics such as quick and confident task performance, a high level of domain knowledge, the ability to recognize spatial and otherwise structured meaningful patterns, and an aptitude for seemingly always knowing what to do in any situation. Regarding this last ability, up to now, an equally important, and possibly even more important, counterpart has been ignored entirely both in CAD research and application and in CAD education. This vital additional ability consists of knowing what not to do in any situation to avoid mistakes, a kind of negative component within strategic knowledge. There exist no catalogues or reference lists containing entries for errors and mistakes committed along with descriptions of how they came about or how they could have been avoided. There exist no frameworks to deal with such kinds of knowledge, and there are no elements in current CAD curricula that try to systematically approach such knowledge or provide any means for students to develop such kinds of expertise. It is within such a scenario that research described in this paper aims to make a fundamental contribution to the improvement of CAD education by providing a means of systematically developing negative expertise. Details of the approach and framework developed to handle negative knowledge and the concepts and structures designed to translate this approach into practice within the context of MCAD are presented and discussed in the following section.

3 APPROACH AND FRAMEWORK

3.1 Negative Knowledge and Expertise

In general, expertise consists of acquired skills and knowledge in a specific domain (cf. [10]), whereas competency, the ability to do something well, can be seen as any acquired skill or knowledge that contributes an essential component for performance or achievement in a given domain. In [9] competence is conceived as the trivial or non-trivial ability to solve problems. Performing efficiently

while committing almost no serious mistakes, i.e. knowing how to avoid grave errors and approaches which are inefficient in certain situations, is an essential feature of professional engineering expertise. This knowing what not to do in certain situations is attributed to knowledge referred to as *negative knowledge*.

Theoretical foundations and early concepts of negative knowledge, unfortunately still neither widely recognized nor seriously considered, can be traced back to work in three different fields. In artificial intelligence, Minsky argues, in his work on negative expertise (cf. [19,20]), that a great deal of what experts know about how to achieve goals and how to avoid disasters lies in knowing about what can go wrong in their domain and which actions might cause trouble and are thus better avoided. He also points out the non-behavioral and reflective character of negative knowledge in his discussions on difficulties in assessing it psychologically and developing a computer-based implementation. In education, the work of Oser and Spychinger [21] on the practice of error culture uses a contrastive approach to define negative knowledge as a type of knowledge that relates to information on false facts and inappropriate action strategies. This approach can be seen as pointing towards negative knowledge as a form of meta-knowledge revealing a regulative impact on positive knowledge. In the examples discussed in their work, the authors also stress the importance of practical experience within a concrete work context, as that is the primary method of obtaining negative knowledge (see also [14]). In knowledge management, the work of Parviainen and Eriksson [24] focused on the declarative aspect of negative knowledge, the *knowing what not to know*, which is in contrast to the by nature more procedural aspect of *knowing what (not) to do*. In their work they distinguished two types of not-knowing relating to the informed and uninformed methods of an individual lacking knowledge relevant to expertise. This distinction addresses in the former case an awareness by the individual of his lack of relevant knowledge, while the latter case supposes both a lack of relevant knowledge and a lack of awareness of this very fact (see also competence levels in [9]). More details on the declarative and procedural aspects of negative knowledge can be found in a recent work by Gartmeier et al. [7]. This work discusses relationships with meta-cognition, and the epistemic potential to enable new insights into various knowledge-related and learning-related fields. It also considers the support given to improving certainty in how to proceed in a task, to increasing efficiency during performance, and to enhancing the depth and quality of reflection on actions and performance.

3.2 Conception of Negative Knowledge

In both engineering science and applied engineering, each system, service, process and product developed can be described by properties known from concepts relating to domain knowledge and experience involved in its design. Of course, there are many additional properties of those artifacts in the real (physical/life) world which we are not aware of now or have no natural or technical means of accessing yet. However, for the purposes of (re-)cognition, reasoning and goal/action building, these properties and their relationships with each other within our limited knowledge are sufficient.

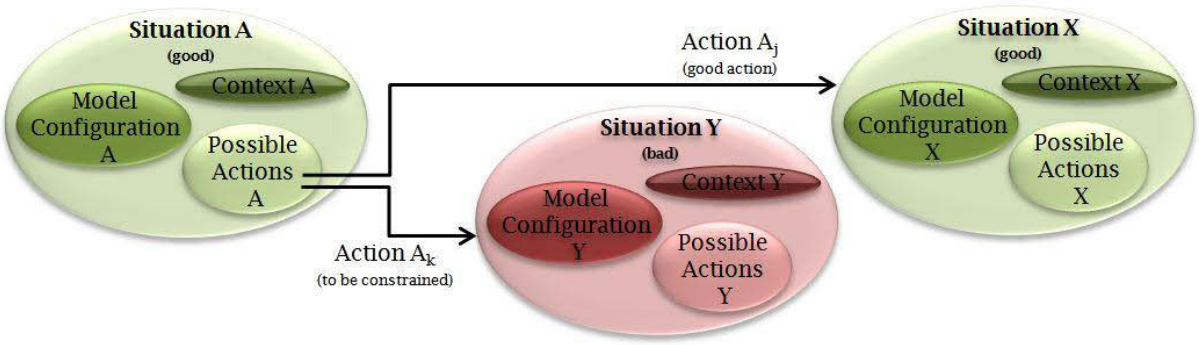

Fig. 1: Overview of the conception of negative knowledge as action constraint.

Now, if we relate our engineering model in its broader meaning and context (cf. detailed discussion on the set of engineering models in the following sub-section) to a concrete set of attribute values associated with well-known concepts sufficiently describing state and properties at any point in time, we shall soon realize that not every constellation, i.e. model configuration, is desirable for our purpose. For example, we might find that model configurations within a computer-based digital product model created during several computer-aided design sessions feature undesirable properties, such as extreme attribute values leading to a possible failure in the function of the product. Alternatively, we might discover that certain configurations violate regulations and/or requirements during the manufacture or usage of the product. These are surely situations that are undesirable, and had thus better be avoided. From a theoretical point of view, desirable situations, indicated by what is considered a good model configuration within a given context employing normative knowledge, represent a reduced set of all possible situations (desirable/undesirable). Here the nature of similarity of desirable situations is determined by reducing variety, which in turn is realized by avoiding undesirable situations by means of restricting actions that have a high tendency (according to what we know and believe to be true) to lead to them. Hence, negative knowledge in terms of knowing what not to do in a certain situation can be conceptualized as a form of action constraint (see Fig. 1). It limits the variety of situations, and consequently their number, by preventing actions that might result in constellations (model configurations) considered not good, i.e. situations deemed undesirable. This concept now features both a quantitative and a qualitative method of determining similarity as an overall defining structural property of situations which are considered desirable. Although, in engineering, quantitative descriptions are often preferred for activities such as simulation or testing and verification, for tasks involving heuristic approaches, knowledge-driven analysis and cognitive activities, a qualitative description is sometimes of more value.

3.3 Concepts and Structures of the Novel Approach

Each engineering task can be abstracted as a list of processes related to individual goals and sub-goals. Each process consists of individual process steps, where an action can be executed in a particular situation, usually changing that situation. A situation can be abstracted as a set of relations associated with particular sets of model configurations and action constraints. This concept of a situation is, in turn, defined by the model and the context (cf. [23]). For instance, a concrete situation is determined by the actual model configuration in a given context. Properties that define the quality of the configuration, i.e. whether the model is good or not, are related to the normative knowledge of an engineering discipline. Model configurations considered good are further sub-divided into more specific configurations such as optimal, efficient and sufficient. Model configurations considered not good are further sub-divided into more specific ones such as dangerous, inefficient and technically outdated.

Of particular interest are significant model configurations, which describe a model configuration in a certain context that is significant in respect to action constraints, which in turn are associated with individual actions. These significant model configurations can be related through a mapping to concrete constraints limiting the actions possible in a particular situation. Such constraints will also prevent transitions that alter properties. In addition, they will attribute values defining a good model to certain properties, and they will attribute other values defining a model as not being good to certain other properties. In this framework, action constraints consist of sub-structures that may be either implicit or explicit in nature. These constraints provide a concept that takes into account the portion of negative knowledge, mostly tacit in nature, which relates to action constraints spanning various different types of situations. Within this framework, a model is assumed to be the set of all engineering models, such as the human-computer interaction model, the CAD model, and the manufacturing model, considered when building associations between model properties, concrete attribute values and significant model configurations. The context is assumed to be a multi-dimensional structure capturing context-related aspects known to be relevant from the domain knowledge, the field of application, the engineering task, etc.

A modified version of an engineering model of human performance, generally used for observation of human-computer interaction, is integrated into the framework. The purpose of this model is to represent components of strategic knowledge that relate not only to positive expertise, but also to negative expertise. It assumes the role of a type of specialized human information processor,

and is referred to hereafter as *GAPACS* (Goals, Actions, Processes, Attributes, Conditions, and Selection rules). This conceptual model facilitates the reduction of a user's interaction with a system such as a computer to its elementary cognitive, perceptual and physical actions, and their relationships to specific user goals and methods used to accomplish those goals. It can be used to both formally express and analyze knowledge of how to select methods and execute actions to accomplish goals related to specific tasks. The basic structure and concepts of the GAPACS model as shown in Fig.2 are derived from the plain GOMS model developed by Card, Moran and Newell [3] and are as follows.

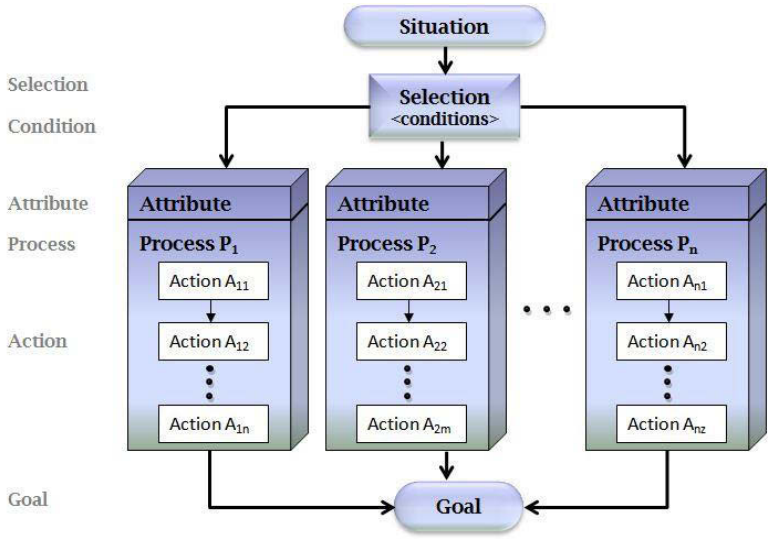

Fig. 2: Concepts and relationships of the GAPACS model.

Goals define what the user wants to accomplish. Actions are performed to reach those goals. Processes represent a method of doing something goal-oriented by defining a proper sequence of actions to actually accomplish a goal. This represents knowledge a user must have to execute a strategy. Attributes such as '*do*' or '*do not*' indicate knowledge of whether a method should be executed as a process within a given situation or not, i.e. should remain constrained under given circumstances. Selection rules state when to use a particular strategy regarding a particular situation. Conditions associated with selections further detail conditions that make a situation significant within the framework of negative knowledge as introduced earlier in this paper. Note that due to the flexibility of the concepts in the original design of GOMS, different levels of abstraction regarding processes and actions can be specified within GAPACS by considering actions within one process as being the goal in a different process and vice versa.

4 APPLICATION AND EXAMPLES FOR MCAD EDUCATION

4.1 Outline

The basic objective of the framework for negative knowledge was to aim for more similarity, which means reducing variety. This objective was approached by formulating negative knowledge as an element of strategic knowledge constraining actions within critical situations that would otherwise lead to errors and mistakes. In other words, the objective was to restrict actions that induce situations best avoided. In the context of MCAD education and negative expertise, as discussed earlier in this paper, this translates into the goal of supporting the development of skills and know-how aimed at providing for the creation of MCAD models containing fewer undesirable structural elements. This can be achieved by systematically reducing model shortcomings introduced by errors and mistakes usually committed by novices, but never by domain experts.

In order to translate this approach, and the novel framework developed, into practice within a given context, as a first step normative knowledge of concepts that characterize the shortcomings of MCAD models needs to be established. This represents a modus operandi that is consistent in nature with negative knowledge, though different from traditional approaches with positive knowledge, where the focus is on efforts to characterize what is considered good, such as, for example, good design practice. For that purpose, by taking into account parametric feature-based CAD systems and the potential of feature technology, the concept of feature deficiency has been developed. This concept can be seen as one important element supporting definition and evaluation of what is to be avoided in respect to particular situations and contexts. Of course, in a different context, for example in computer-aided design for environmental conscious products or computer-aided architectural design, such a concept would have a completely different emphasis.

To illustrate the application of this approach employing the framework of negative knowledge, a small carefully chosen selection of critical situations, and how they are defined in terms of model parameter attributes and feature deficiency, is presented. The selection represents a portion of currently ongoing analysis and compilation efforts based on MCAD models, protocols, and questionnaires obtained during CAD laboratory activities in the department which the authors represent. It has been collected over the past academic year. The parametric feature-based CAD system used in class-based education and in practical CAD laboratory exercises is the commercially available *Solid Edge* CAD system, a core component of the Velocity Series™ portfolio distributed by Siemens AG.

4.2 The Concept of Feature Deficiency

Within the context of parametric feature-based CAD systems, feature technology provides a promising leverage to implement the design rationale in a modeled object, i.e. capture and convey design intent. This can be achieved by relating a shape and its topological and geometric characteristics to the structure of features and their selection, order, and organization. Within the scope of this work, feature technology is used as both a means to support learning and implementation of design intent in CAD models (within the larger context of systematically developing negative expertise) and as an anchor concept to evaluate CAD models in respect to missing/poor design intent. To achieve the latter, the concept of *feature deficiency* has been developed and integrated into the framework. Feature deficiency is used as a qualitative measure to help express certain characteristics of situations during modeling. These characteristics usually lead to models being difficult to understand and poorly structured (failing to convey the intent of the design) and are thus better avoided. Feature deficiencies are sub-divided into two groups regarding their definition/creation and the actual evaluation/computation of the features as implemented in most commercially available parametric feature-based CAD systems. This distinction is between *basic feature deficiencies* and *deficient feature sequences*. Individual feature deficiencies as used within the framework application are defined as follows.

Basic feature deficiencies

Deficient feature type:

The feature type does not capture the design intent. For example, a cutout feature type has been used to model a hole. This represents a situation which will eventually lead to feature parameters being missing, hence making an alteration to the model less intuitive. For example, a simple hole can be changed into a counter-bore hole just by changing the type of hole in the hole feature parameters. However, if the hole feature is not available in the CAD model feature list, the execution of this alteration will be more difficult and error prone.

Deficient feature shape:

The shape chosen for a feature is inappropriate for modeling the intended feature. For example, a blind hole that contains a flat bottom is inconsistent with technological constraints in respect to manufacturing the hole with a drill bit.

Deficient 2D feature profile:

The profile chosen is inappropriate for modeling the intended feature. For example, an extrusion with a profile made of several disjoined loops will result in a pattern instead of a single feature.

Deficient associative feature geometry:

The geometry used to create relationships between features has a high risk of instability. For example, an edge employed as a geometric entity associated with a round command could disappear in the model.

Deficient 2D feature constraint:

Two-dimensional constraints related to the profile are not sufficient to preserve the geometric structure of the profile after a model alteration takes place. For example, a rectangle could change into a trapezium.

Deficient 3D feature constraint:

Limits of extrusions/depressions do not correctly represent the original design intent. For example, a through cutout has a limit on its fixed depth as shown in Fig. 3.

Fig. 3: Example of a through cutout with a fixed depth limit.

Deficient 2D feature dimension:

The feature profile does not have sufficient dimensions to allow for consistent model alterations. For example, a loop feature profile without the dimension for length cannot be altered in length.

Deficient 3D feature dimension:

The height, depth, width or length of a feature does not reflect the design intent. This represents a deficiency which is usually associated with a redundant feature sequence (see detailed description below).

Deficient feature sequences

Redundant feature sequence:

A single functional/technological feature is modeled with several separate feature commands. For example, a triangular rib feature is modeled using a sequence consisting of a rectangular protrusion and a cutout as shown in Fig. 4.

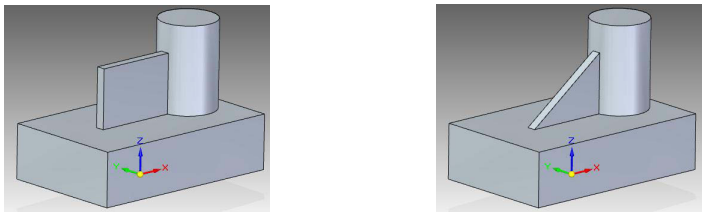

Fig. 4: Examples of a redundant feature sequence for the modeling of a stiffening rib.

Removing feature sequence:

Feature geometry is removed from the model by using another feature that spatially covers the first feature's volume, thus obliterating it. For example, the shape aspect as geometry related to a depression of a hole feature is deleted in the model by adding a feature that has a shape aspect with geometry related to an obliterating extrusion.

Destructive feature sequence:

Feature geometry is altered to the point where its functional/technological meaning (semantics) is lost. For example, depression-related features such as holes become partially covered at their openings as shown in Fig. 5(a) or have added additional openings as shown in Fig. 5(b,c).

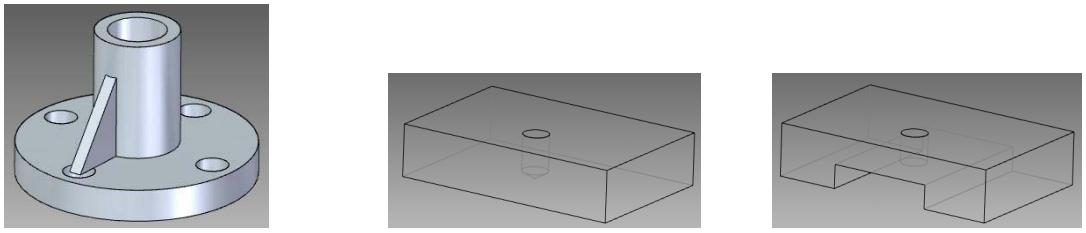

Fig. 5: Examples of a destructive feature sequence. From left to right: (a) A rib feature covering the entrance of a hole feature, (b) A blind hole feature, (c) A blind hole feature being changed into a through hole feature by adding a slot feature.

Unstable feature sequence:

Features are linked to each other in such a way that if a feature is removed from the feature list, then model re-evaluation/generation leads to features in the feature sequence with broken/unstable links. For example, deleting the chamfer feature as shown in Fig. 6(a) will result in an unstable feature sequence as shown in Fig. 6(b).

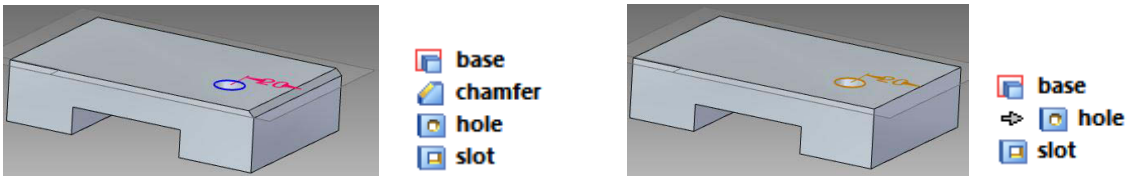

Fig. 6: Example of an unstable feature sequence. From left to right: (a) Original feature sequence, (b) Unstable feature sequence where the chamfer feature is removed.

4.3 Examples from MCAD Practice

In the following a selection of practical examples encountered during the analysis of material from empirical results is presented and discussed. The selection represents examples of significant model configurations that are based on different combinations of feature deficiency encountered as typical cases in CAD practice in both industrial settings and university laboratory classes. To improve readability and transparency, in the following all examples are presented using a basic text-based notation instead of a formal notation employing model parameters and attribute value settings.

Situation A:

Significant Model Configuration	
GAPACS Model:	Goal is to model a counter-bore through hole
	Selecting a hole feature command is available as an action
Geometric Model:	The hole reference plane is an available entity of the model
Action Constraint	Do not use a revolved cutout to model the counter-bore through hole
Anticipated Feature Deficiency	
Deficient Feature Type:	The feature of feature type hole is missing from the feature list
Deficient 2D Feature Profile:	The profile of the shape results in a poly-line as shown in Fig. 7(a) that does not reflect the profile typical for a correctly modeled hole of this type. It should instead be based on a circular profile (see Fig. 7(b))
Deficient 2D Feature Dimension:	The dimensions of the hole diameters are missing, due to the improper shape definition of the profile that is being used

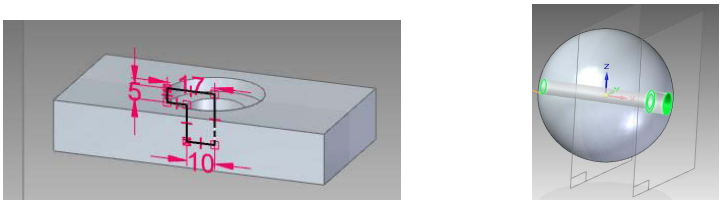

Fig. 7: Examples of feature deficiency for Situation A. From left to right: (a) Counter-bore hole feature modeled with a revolved cutout command, (b) Modeling situation where the geometric entities of the hole feature reference planes are not geometric entities (faces) of the model.

Situation B:

Significant Model Configuration	
GAPACS Model:	Goal is set to model a blind hole
	Selecting a hole feature command is available as an action
Geometric Model:	The hole reference plane is an available entity of the model
Manufacturing Model:	The range of a drill bit diameter exists for the hole diameter
Action Constraint	Do not use an extruded cutout to model the blind hole
Anticipated Feature Deficiency	
Deficient Feature Type:	The feature of feature type hole is missing from the feature list
Deficient Feature Shape:	A flat ended hole is deficient in respect to the shape that a drilled blind hole will have (see Fig. 8(a,b))
Deficient 3D Feature Dimension:	The finite depth parameter of the (extruded) cutout does not correctly describe the actual hole depth, because the conical tip is not taken into account (see Fig. 8(a,c))
Anticipated Feature Sequence Deficiency	
Redundant Feature Sequence:	In order to model the conical tip, an additional command will be required

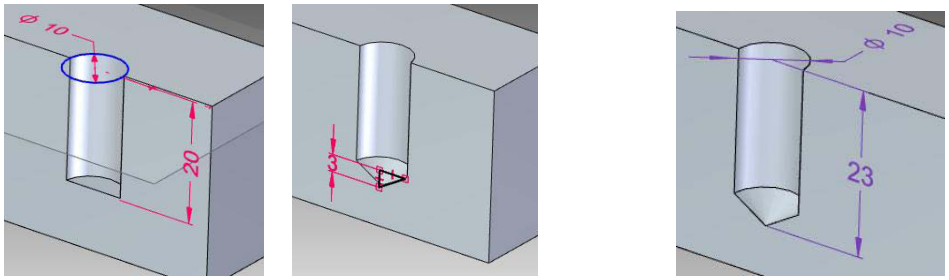

Fig. 8: Examples relating to feature deficiency for Situation B given in sectional view. From left to right: (a) Blind hole feature modeled with an extruded cutout command, (b) Blind hole feature with a conical tip modeled with a revolved cutout command, (c) Blind hole modeled with a correctly related hole feature.

Situation C:

Significant Model Configuration	
GAPACS Model:	Goal is to model a rectangular cutout
	Selecting a pocket feature command is not available as an action
Geometric Model:	The pocket reference plane is an available entity of the model
	The geometric structure of the pocket has to remain unchanged
Action Constraint	Do not use an under-constrained profile
Anticipated Feature Deficiency	
Deficient 2D Feature Constraint:	The under-constrained rectangular profile (Fig. 9(a)) may be converted into a geometrically different shape (e.g. a quadrilateral such as the trapezium as shown in Fig. 9(b)) after a change in value of one of its defining dimensions

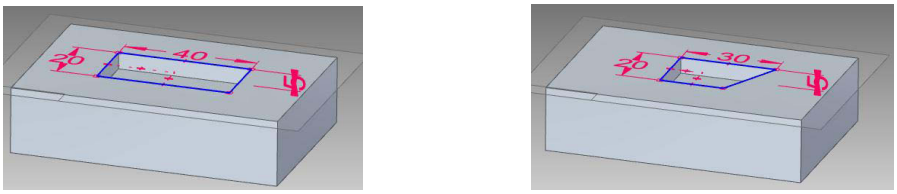

Fig. 9: Example of feature deficiency for Situation C. From left to right: (a) The rectangular under-constrained profile of a pocket before the dimension is altered, (b) The rectangular under-constrained profile of the pocket after the dimension has been altered.

Situation D:

Significant Model Configuration	
GAPACS Model:	Goal is to assign a depth value within the modeling of a through cutout
	Selecting a through cutout command is available as an action
Geometric Model:	The cutout reference plane is an available entity of the model
	The cutout surface is an available entity of the model
Action Constraint	Do not assign a fixed value to the cutout depth
Anticipated Feature Deficiency	
Deficient 3D Feature Constraint:	If the thickness of the base material subject to cutting is increased in the re-design, the through cutout as modeled and depicted in Fig. 10(a) will turn into a blind cutout as shown in Fig. 10(b)
Deficient 3D Feature Dimension:	The depth value in the model does not represent the actual depth of the cutout as designed (see again Fig. 10(a,b))

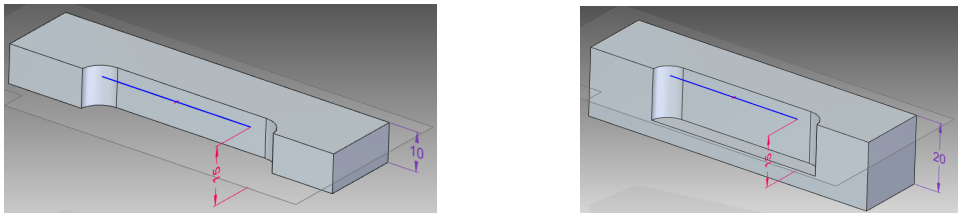

Fig. 10: Example of feature deficiency for Situation D shown in sectional view. From left to right: (a) The through loop feature of the through cutout with a fixed depth, (b) The through loop feature of the cutout turning into a blind (loop) cutout.

5 CONCLUSIONS AND FUTURE WORK

The concepts and structures of a novel approach aimed at facilitating the development of negative expertise within MCAD education have been presented and discussed. Examples were given to illustrate central concepts of the framework and how they relate to and interact with domain knowledge application and skill development. In particular, the evaluation of characteristics that determine situations to be avoided was based on the concept of feature deficiency and the related creation of deficient CAD models within the application field of mechanical engineering. This new approach, which is based on the concept of negative knowledge as an action constraint limiting the action space in respect to a critical situation, which in turn is determined by significant model configurations, bears the potential to improve shortcomings in both industrial practice and academia. Such shortcomings are related to an inability to use advanced CAD systems as communication tools to convey design intent, and to the creation of poorly structured CAD models that are difficult to understand and alter.

Within current CAD courses for mechanical engineering students, offered by the department represented by the authors, several new elements have now been included on an experimental basis. These new elements are in the form of short course units on feature technology and on how to develop negative expertise employing aspects of the novel framework of negative knowledge and the concept of feature deficiency outlined above. Such teaching is aided by brief catalogues which have been compiled, and by annotated examples. The courses are accompanied by questionnaires to be filled out on a voluntary basis by students at the end of a semester. These efforts currently underway are aimed at increasing understanding and insight on a theoretical and empirical basis regarding the design and implementation of modified instruction methods, modeling examples, and laboratory exercises later to be included permanently in a properly revised MCAD curriculum.

REFERENCES

[1] Bidarra, R.; Bronsvoort, W. F.: Validity Maintenance of Semantic Feature Models, In Proceedings of the ACM Symposium on Solid Modeling and Applications, June 08-11, Ann Arbor, MI, USA, 1999, 85-96. http://dx.doi.org/10.1145/304012.304021

[2] Bhavnani, S. K., Reif, F., John, B. E.: Beyond Command Knowledge: Identifying and Teaching Strategic Knowledge for Using Complex Computer Applications, CHI 2001, ACM Conference on Human Factors in Computing Systems, CHI Letters, 3(1), 229-236.

[3] Card, S.; Moran, T. P.; Newell, A.: The Psychology of Human Computer Interaction, Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1983.

[4] Chester, I.: 3D-CAD: Modern Technology – Outdated Pedagogy?, Design and Technology Education, 12(1), 2007, 7-9.

[5] Chester, I.: Teaching for CAD Expertise, International Journal of Technology and Design Education, 17(1), 23-35. http://dx.doi.org/10.1007/s10798-006-9015-z

[6] Danjou, S.; Lupa, N.; Koehler, P.: Approach for Automated Product Modeling Using Knowledge-Based Design Features, Computer-Aided Design and Applications, 5(5), 2008, 622-629, DOI:10.3722/cadaps.2008.622-629. http://dx.doi.org/10.3722/cadaps.2008.622-629

[7] Gartmeier, M.; Bauer, J.; Gruber, H.; Heid, H.: Negative Knowledge: Understanding Professional Learning, Vocations and Learning, 1(2), 2008, 87-103. http://dx.doi.org/10.1007/s12186-008-9006-1
[8] Hartman, N. W.: Defining Expertise in the Use of Constraint-Based CAD Tools by Examining Practicing Professionals, Engineering Design Graphics Journal, 69(1), 2005, 6-15.
[9] Heid, H.: Erfordernis und Problematik einer Unterscheidung zwischen Verhalten und Verhaltensdisposition, in: Beck, K., Müller, W., Deißinger, T. and Zimmermann, M. (eds.), Berufserziehung im Umbruch, Deutscher Studienverlag, Weinheim, Germany, 1996, 79-85.
[10] Hunt, E.: Expertise, Talent, and Social Encouragement, in: Ericsson, K. A., Charness, N., Feltovich, P. J. and Hoffman, R. R. (eds.), The Cambridge Handbook of Expertise and Expert Performance, Cambridge University Press, New York, NY, USA, 2006, 31-38. http://dx.doi.org/10.1017/CBO9780511816796.003
[11] Iyer, G. R.; Mills, J. J.: Design Intent in 2D CAD: Definition and Survey, Computer-Aided Design and Applications, 3(1-4), 2006, 259-267.
[12] Jagenberg, J. T.; Gilsdorf, E. A.; Anderl, R.; Bornkessel, T.: Knowledge Driven Design Features for the Product Life Cycle of Engine Parts, In Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, August 30 – September 2, San Diego, CA, USA, Vol. 8, 2009, 721-728.
[13] Johnson, D. J.; Diwakaran, R. P.: An Educational Exercise Examining the Role of Model Attributes on the Creation and Alteration of CAD Models, Computers and Education, 57(2), 2011, 1749-1761. http://dx.doi.org/10.1016/j.compedu.2011.03.018
[14] Kolodner, J.: Towards an Understanding of the Role of Experience in the Evolution from Novice to Expert, International Journal of Man-Machine Studies, 19(5), 1983, 497-518. http://dx.doi.org/10.1016/S0020-7373(83)80068-6
[15] Kyprianou, M. R.: Shape Classification in Computer Aided Design, Ph.D. Dissertation, University of Cambridge, Cambridge, U.K., 1980.
[16] Lang, G. T.; Eberts, R. E.; Gabel, M. G.; Barash, M. M.: Extracting and Using Procedural Knowledge in a CAD Task, IEEE Transactions on Engineering Management, 38(3), 1991, 257-268. http://dx.doi.org/10.1109/17.83758
[17] Mandorli, F.; Cugini, U.; Otto, H. E.; Kimura, F.: Modeling with Self Validation Features, In Proceedings of the ACM Symposium on Solid Modeling and Applications, May 14-16, Atlanta, GA, USA, 1997, 88-96, DOI:10.1145/267734.267757 http://doi.acm.org/10.1145/267734.267757.
[18] Ma, Y.-S.; Tong, T.: Associative Feature Modeling for Concurrent Engineering Integration, Computers in Industry, 51(1), 2003, 51-71. http://dx.doi.org/10.1016/S0166-3615(03)00025-3
[19] Minsky, M.: The Society of Mind, Simon and Schuster, New York, NY, USA, 1986.
[20] Minsky, M.: Negative Expertise, International Journal of Expert Systems, 7(1), 1994, 13-19.
[21] Oser, F.; Spychiger, M.: Lernen ist schmerzhaft: Zur Theorie des negativen Wissens und zur Praxis der Fehlerkultur, Beltz, Weinheim, Germany, 2005.
[22] Otto, H. E.: From Concepts to Consistent Object Specifications: Translation of a Domain-Oriented Feature Framework into Practice, Journal of Computer Science and Technology, 16(3), 2001, 208-230. http://dx.doi.org/10.1007/BF02943200
[23] Otto, H. E., Mandorli, F.: Negative Knowledge and Eco-Design, In Proceedings of the International Conference on EcoBalance, November 20-23, Yokohama, Japan, Paper-No. P-133, 2012.
[24] Parviainen, J.; Eriksson, M.: Negative Knowledge, Expertise and Organisations, International Journal of Management Concepts and Philosophy, 2(2), 2006, 140-153.
[25] Peng, X., McGary, P., Johnson, M., Yalvac, B., Ozturk, E.: Assessing Novice CAD Model Creation and Alteration, Computer-Aided Design and Applications, PACE(2), 2012, 9-19, DOI: 10.3722/cadaps.2012.PACE.9-19. http://dx.doi.org/10.3722/cadaps.2012.PACE.9-19
[26] Rynne, A.; Gaughran, W.: Cognitive Modeling Strategies for Optimum Design Intent in Parametric Modeling, Computers in Education Journal, 18(3), 2008, 55-68.
[27] Shah, J.; Mantyla, M.: Parametric and Feature Based CAD/CAM: Concepts, Techniques, and Applications, John Wiley & Sons, New York, NY, USA, 1995.
[28] Wiebe, E. N.: Transfer of Learning between 3D Modeling Systems, Engineering Design Graphics Journal, 67(3), 2003, 15-28.