# Chapter 17
# KBE-Modeling Techniques in Standard CAD-Systems: Case Study—Autodesk Inventor Professional

**Paul Christoph Gembarski, Haibing Li, and Roland Lachmayer**

## 17.1 Introduction

Enhanced use of carry-over-parts, shorter product life cycles and product customization lead to a frequent modification and the adaptation to new functional or design requirements of digital product models [1]. Basis for this is the ability of parametric modeling in today's CAD-systems which is the use of variable values (parameters) for dimensions and variable constraints between objects and models in a CAx-system [2].

Knowledge-Based Engineering (KBE) extends this approach in order to implement explicit design knowledge into the virtual product model [3]. The overall goal is to transform a design problem into a configuration problem using, e.g., the link to dimensioning or calculation formula, design rules or manufacturing restrictions.

### *17.1.1 Motivation*

The product development process is a structured sequence of creative activities, which aims at translating technical and design requirements into a product specification with its corresponding geometric models. On the one hand, the exploration and limitation of the possible solution space is dependent of the designer's experience and his ability to make design knowledge explicit. The answer to the question why a product looks the way it actually does, has not only to be answered but documented. On the other hand different steps in the design process contain various routine tasks.

P.C. Gembarski (✉) • H. Li • R. Lachmayer
Institut für Produktentwicklung und Gerätebau, Leibniz Universität Hannover,
Welfengarten 1a, Hannover 30167, Germany
e-mail: gembarski@ipeg.uni-hannover.de; li@ipeg.uni-hannover.de;
lachmayer@ipeg.uni-hannover.de



J. Bellemare et al. (eds.), *Managing Complexity*, Springer Proceedings
in Business and Economics, DOI 10.1007/978-3-319-29058-4_17

According to Verhagen, one objective of KBE is to reduce time and cost of product development, which is generally realized through automation of repetitive design tasks as well as capture and re-use of design knowledge [4]. In business models like mass customization, where a product tailored to a customer's needs has to be developed and manufactured with mass production efficiency, the demand for such supporting methodologies and the corresponding modeling techniques is high. One instance would include, e.g., technical product configurators or design configurators. Nevertheless, different authors report that KBE still has not achieved a considerable breakthrough beside different automotive and aerospace applications [5].

Different aspects of KBE have already been discussed for decades, e.g., the product configuration paradigms [6, 7] or process models for creating KBE applications like MOKA and KNOMAD [4]. Other contributions aim at pointing out the delimitations to other research fields such as knowledge engineering and knowledge management. Regarding the impact on product modeling and virtual prototyping, there exist only a little number of contributions, which aims at establishing theoretical foundations for KBE. Most authors do not present concrete design methodologies, modeling principles, or detailed application examples. To date, no scientific books can be found that are dedicated to this topic.

We will bridge a part of this gap by showing how different methods and functionalities, that are already implemented and accessible in the standard package of the CAD-system Autodesk Inventor Professional, can be used for setting up virtual prototypes and their solution spaces as KBE models.

### *17.1.2 Structure of the Paper*

The following Chap. 2 provides the theoretical background for KBE with regard to geometry manipulation and reasoning techniques. In Chap. 3 different modeling techniques in the CAD-system Autodesk Inventor Professional are exemplarily introduced and discussed in context of KBE. Chapter 4 then presents different application examples. Closing the paper, Chap. 5 contains a brief summary and drafts further research questions for future studies.

## 17.2 Theoretical Background

According to Chapman et al., "KBE represents an evolutionary step in computer-aided-engineering (CAE) and is an engineering method that represents a merging of object-oriented programming (OOP), artificial intelligence (AI) and computer-aided-design (CAD) technologies, giving benefit to customized or variant design automation solutions" [8] (Fig. 17.1).

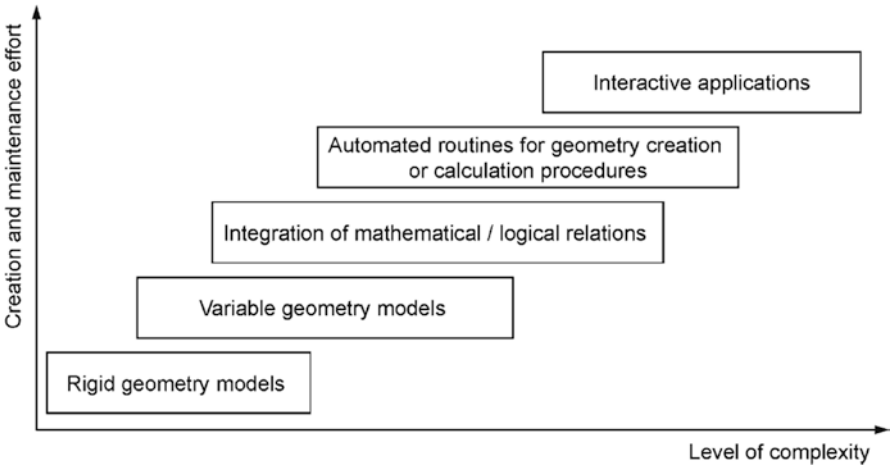


**Fig. 17.1** Different types of knowledge-based design applications according to Hirz [3]

Hirz emphasizes that "knowledge-based design supports design processes by reusing predefined methods, algorithms or results, and it is integrated into specific tasks or workflows that are involved in the design processes" [3].

### *17.2.1 Parametric Design*

Basis for knowledge-based design is the application of parametric CAD. There are three major benefits from using parametric design in opposite to rigid geometry [9]:

1. Automatic change propagation
2. Geometry re-use
3. Embedding of design/manufacturing knowledge with geometry

It is commonly accepted that the parameterization of a virtual prototype leads to the individual description of the geometry and its defining parameters and constraints. According to Vajna basically four different parameter types have to be differentiated [10]:

- *Geometric parameters* define the shape of a part or assembly. To these belong all kinds of dimensions and positioning constraints.
- *Topology parameters* have to be understood as structural parameters which can control, e.g., the suppression state of a component in an assembly or that of a feature in a part model
- *Physical parameters* determine the physical properties of the design
- *Process or technological parameters* contain, e.g., manufacturing restrictions like minimum bend radiuses or the angle of mold release slopes for cast designs.

A model's parameters are linked by arithmetical or logical constraints. Another class of constraints is geometric ones like setting two sketch lines parallel to each other or placing a component's connection point coincident on the origin of an assembly.

### *17.2.2 Parameter Control*

Defining mathematical relations between parameters offers the possibility of differentiating leading and driven parameters. Thus, the designer not only models the geometry but also had to plan the configuration concept and the parameterization of the specific component he is drafting.

The use and integration of dimensioning formula and equations in the CAD model supports automated geometry definition and change propagation [3]. Prerequisite is that the CAD system has the ability to create (user) parameters not only for length or angular dimensions, but also supports calculation and processing of all other kind of units, e.g., for stresses, forces, or moments of inertia.

The more complex the component, the more complicated may be the configuration concept, which calls for structuring parameters at different levels. Therefore, assemblies can include a skeleton model, which defines component positioning or superordinate geometrical characteristics, e.g., based on the structural design [11]. The corresponding parameters either can be transferred via design rules within the top-level assembly or exported into the respective part models, which establishes a permanent data link between skeleton and part document.

Another way of structuring parameters is the possibility of externalizing the calculation and input of relevant parameters, which then drives the geometry within the CAD model. This can either be done through the import of text files or the link to commercially available spreadsheet software. The latter commonly offers additional mathematical and statistical operations compared to those implemented in the CAD system itself [3]. Another important fact is that relevant data for the definition and specification of components can be stored on different worksheets and then be linked by use of matrix-operations like VLOOKUP in MS Excel. Such a formulated knowledge base has to be understood as significant element within a CAE environment [12].

### *17.2.3 Design Rules*

The implementation and formulation of design rules strongly depends of the CAD system. Only the minority of systems are able to set up and compute design rules within the functionalities of the standard configuration, most of these systems need extensions like the Knowledge Workbenches for CATIA or Knowledge Fusion for Siemens NX [5].

Basically, the rule concept is grounded upon the IF-THEN-ELSE-notation known from software development. Rules are fired procedurally and can be used to execute subordinate rules or delete them temporarily from the working memory [13].

The rule concept is very well known as reasoning mechanism of the expert systems from the 1980s [6].

### *17.2.4 Intelligent Templates*

An intelligent template has to be understood as a parametric, updatable, and reusable building block within a digital prototype. For specific components, templates include all necessary design rules and features. So, templates can support the collection of expert knowledge and integrate existing knowledge from former development projects into the current design process [3].

According to Cox the creation of an intelligent template involves four steps [14]:

1. Use past experience, define the boundaries for the solution space
2. Map the product development process backwards into the context of the template
3. Develop a generic parametric model of all necessary products and artifacts
4. Map the specific model parameters into a common set of configuration parameters

### *17.2.5 Automation Routines and Macros*

According to Hirz, "the ability to create macros can be very helpful for enabling automatic sequences of features and actions" [3]. Two approaches have to be distinguished; on the one hand the code can be implemented either internally in the CAD system or in single CAD models and drawings. Depending on the application programming interface (API) and its implementation technology the macros are interpreted row by row, class concepts or inheritance like known from object oriented programming may not be fully available. On the other hand the code can be written externally into a compiled software package which then drives the CAD system remotely. There, all functionalities of the used software development environment can be addressed.

Since the automation within a CAD system strongly depends on the system's functionality itself and the corresponding API model, it will not be considered further in this article.

### *17.2.6 Reasoning Techniques*

A completely different approach to classify KBE methods is with regard to their abilities and problem solving methods in order to derive new configurations based on existing designs. This dates back to the expert systems of the 1980s and 1990s, where similarly to a human expert a knowledge-based system makes use of a reasoning mechanism which is also called inference engine [5].

This implies that all necessary engineering knowledge from all participating experts, be it process and simulation specialists, designers or production engineers, has to be formulated in a domain specific knowledge base, which extends the geometric product model [15].

Basically, the following three different paradigms can be distinguished [6]:

- *Rule-based reasoning*: The knowledge representation relies to design rules like mentioned in Sect. 17.2.3. A major disadvantage of this kind of systems is their lack of separation between domain knowledge and control strategy. As reported by McDermott, this results in bad maintainability when the system exceeds a certain amount of rules [13].
- *Model-based reasoning*: The limitation of the possible solution space is done based upon a physical and/or logical model (constraint-based) or by representation of resource consumption and allocation (resource-based) [16].
- *Case-based reasoning*: In this approach, the knowledge representation is not explicitly modeled in form of rules or constraints. The knowledge necessary for reasoning is stored in cases that represent former configurations. Depending on the degree of maturity of the inference engine the system either is limited to search for existing solutions, which match exactly to a given requirements profile, or the system is able to assort a set of existing cases, which represent the best-fit. Highly developed case based systems are able of mixing or altering exiting cases in order to adapt them to new situations.

Differently than traditional knowledge-based systems, a KBE system shows no crisp separation between knowledge base and inference mechanism. The control strategy for accessing and manipulating the knowledge base and the domain knowledge itself are strongly intertwined [5].

## 17.3 KBE Modeling in Autodesk Inventor

In this chapter different modeling techniques in the CAD-system Autodesk Inventor Professional are exemplarily introduced and discussed in context of KBE. In the following subsection it is evaluated what parameter types as referred in Sect. 17.2.1 can be implemented in Inventor part and assembly models. Afterwards we present possibilities to link these parameters via equations and rules. Using these functionalities dynamic assembly models and intelligent templates can be set up.

The implementation of MS Excel spreadsheets allows the integration of a behavioral model in the background of the geometric one which leads to setting up technical product configurators.

### *17.3.1 Integration of Different Parameter Types*

Within Inventor every parameter regardless of its type (model, reference, or user parameter) has a specific set-up and consists of the four parts: name, value, unit, and comment. Model parameters are introduced by Inventor with every dimension, as

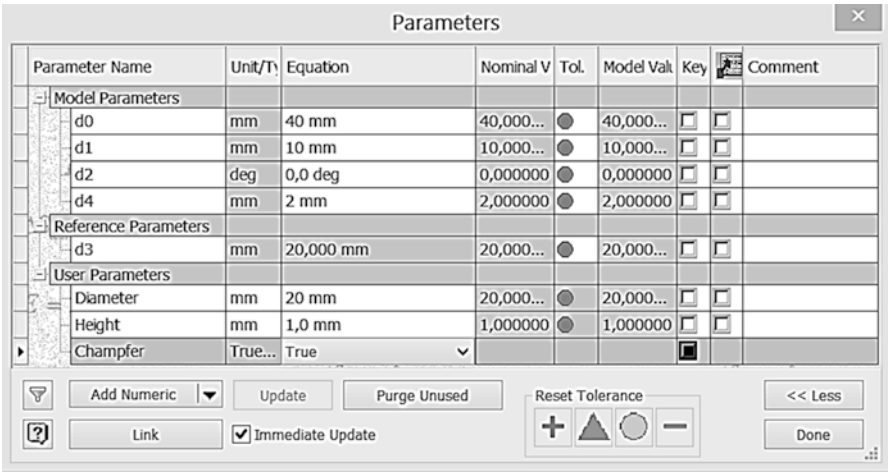


**Fig. 17.2** Inventor Professional 2016: parameter table

well as reference parameters. The latter indicate either a driven dimension (element is over-constraint) or an imported parameter.

All parameters are managed in the parameter table as depicted in Fig. 17.2. In this dialog, the user can change all parameter values including adding names and comments to each of them or set new user parameters. Those can either be numerical or Boolean, a third type of user parameters is text (not shown). If a numerical parameter has to contain a value list instead of a single value, e.g., to choose between certain thicknesses for a sheet metal part, the parameter may be defined as multi-value. The input format then changes from text box to dropdown box.

As unit types Inventor can use basically all physical units with all suitable prefixes. This includes units for length (mm, inch, nautical mile, etc.), angularity (radian, degree) but also for mass, forces, power, velocities, electrical, or luminosity to name only a few categories.

### *17.3.2 Equations and Design Rules*

Setting up an equation within Inventor can either be done directly at dimensioning or centralized in the parameter table. As mathematical operators, all basic arithmetic operations, trigonometric functions, roots and powers as well as rounding operations may be used.

In the following example, the diameter of a bold has to be calculated according to the dimensioning formula, as in (17.1), with $k$ as fitting factor, $K_A$ as application factor, *Force* as applied force, and *Bend stress* as maximum bend stress depending on the material of the bolt:

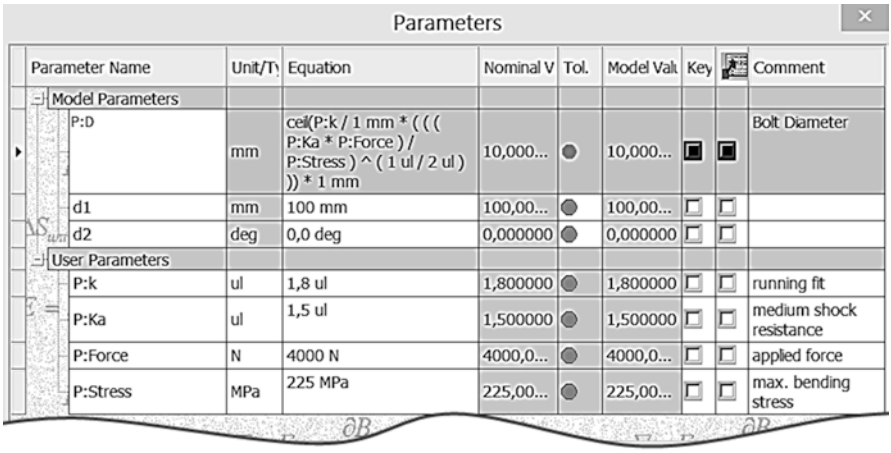


**Fig. 17.3** Dimensioning formula implemented in an Inventor part document

$$d \approx k \sqrt{\frac{K_A \ \text{Force}}{\text{Bendstress}}} \tag{17.1}$$

The equation is entered in the parameter *P:D* which defines the bold diameter as depicted in Fig. 17.3.

Note that it is necessary to cancel down the unit within the ceiling operation since Inventor expects a dimensionless factor which then has to be multiplied by 1 mm again.

Such an equation is computed every time when a rebuild to the model geometry is processed.

Another way of linking parameters is the use of design rules which can be done within the iLogic environment as depicted in Fig. 17.4. The iLogic programming language is similar to script languages. Common constructs like if-then-else or select-case decision trees, while loops, the use of sub procedures and a class concept are usable. As command library the snippets include code templates for almost every modeling context within Inventor.

In the example above, the two parameters *d0* and *d1* are linked to prior defined user parameters via equations. In addition, the suppression state of a chamfer feature is linked to a Boolean parameter. All parameters from each component or feature can be addressed through the model tree within the iLogic rule editing dialog. This is the same for assemblies, where parameters of different parts can be linked to each other (this will be shown in Sect. 17.3.3 in context of dynamic assembly models).

In contrast to parameter equations, iLogic rules are not automatically computed at every rebuild of the model. The computation either has to be triggered manually or linked to certain events like geometry update and closing a file.

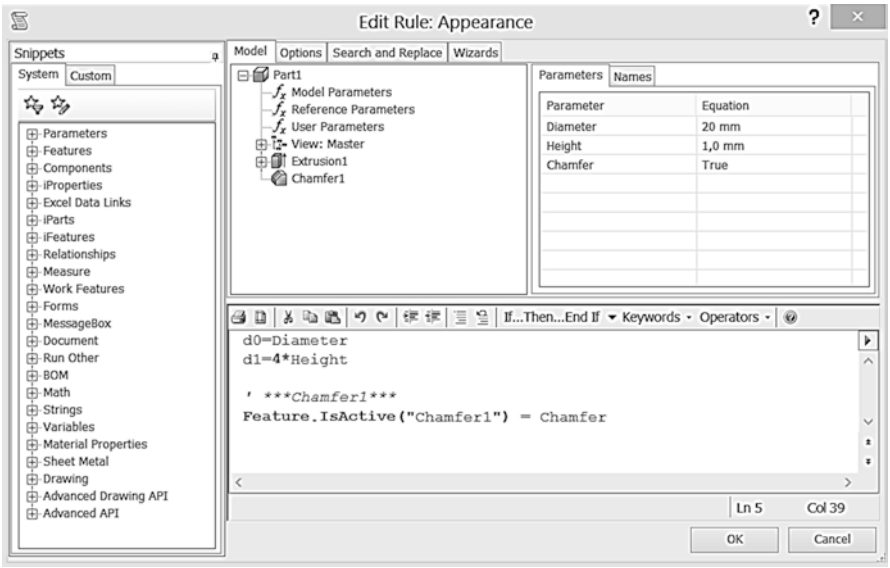


**Fig. 17.4** iLogic rule editor

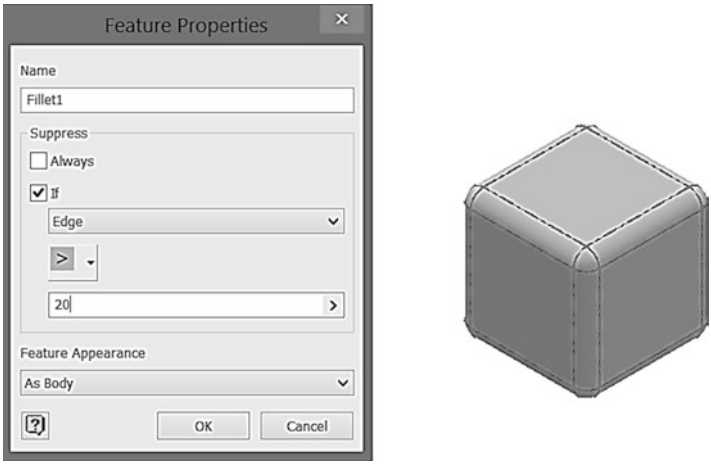


**Fig. 17.5** Suppression state definition in feature properties dialog

Within the part document exists another way of using classical design rules for topological parameters. Therefore, a dimension parameter is linked directly to the suppression state in the feature properties of the feature to be controlled.

In the example shown in Fig. 17.5 the cube's fillet is suppressed when the length of the edge (described in a parameter named *edge*) exceeds 20 mm.

### *17.3.3 Dynamic Assembly Models and Intelligent Templates*

In this subsection we present two different methodologies for modeling of a dynamic assembly model for use as intelligent template. The first approach uses a skeleton behind the geometric models, whereas the second one links the necessary parameters via iLogic rules. The running example for this subsection is the base frame of a pipe support as depicted in Fig. 17.6.

For the skeleton model a part document is created which contains the four basic configuration parameters height over ground of both decks, the width of the rack and the thickness of the flange plates. Additionally the cross section for the used beams and the layout of the flange plate are defined as sketches (Fig. 17.7).

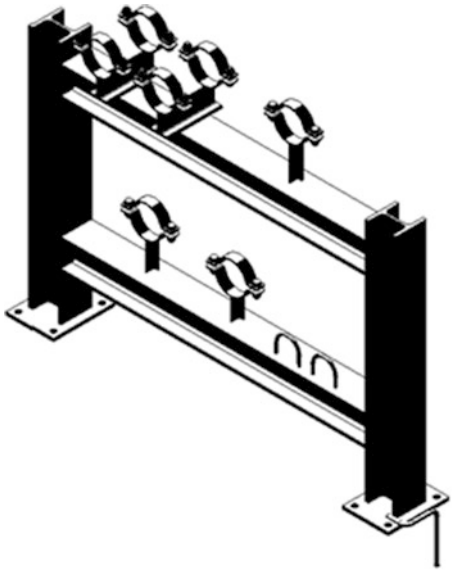

**Fig. 17.6** Pipe support

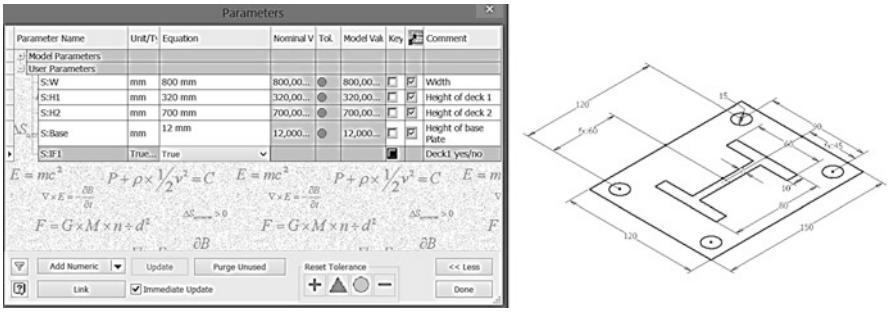


**Fig. 17.7** Skeleton model for the base frame

Via the derive component command the skeleton is imported into the corresponding part documents for the beams and the flange plate. Within the derivation dialog only the necessary parameters and sketches are marked for import (Fig. 17.8).

Afterwards the extrusion for the first beam is set up. Therefore, the length dimension is calculated from the imported parameters. The procedure is similar for the second beam and the flange plate.

The next step is building everything together in the assembly document. Here, the parts are traditionally placed and constrained to each other. To adapt the assembly to new geometric boundaries only the configuration parameters within the skeleton have to be modified (Fig. 17.9).

Advantages of this approach are a short set-up time and the change propagation within the derived part. A change not only to the configuration parameters, but also to the sketch dimensions of cross sections and flange are immediately processed to the part documents then. Disadvantage is the fact that each part document has a permanent data link to the skeleton part what might have a negative effect on rebuild and document loading times. The link can be broken, but an artifact of the skeleton remains in the documents. Removing this would result in rebuild errors. Additionally, the parameterization has to be planned beforehand because all necessary parameters and sketches have to be defined in the skeleton.

The second approach to model the base frame is linking parameters by iLogic rules. Here the beams and the flange plate are modeled and assembled traditionally without skeleton or other knowledge implementation. The next step is to add the configuration parameters to the assembly file and then to append the iLogic rules. This is not completely comparable to the skeleton variant above since the skeleton also controls the dimensions for the beams' cross section. If the dimensions would also have to be driven by iLogic rules, then additional parameters have to be linked (Fig. 17.10).

If the geometry alteration should not be made by change of the user parameters in the parameter table, a graphical user interface can be built as iLogic form like depicted in Fig. 17.11. Here, different input controls, e.g., textboxes, sliders, or checkboxes, can be used. Additionally, when one of the input parameters is multi-value, the user may choose between combo box, list box, or radio buttons. Other controls, e.g., command buttons for running macros, can also be implemented.

The advantage of this method is that for a known model set-up the linking of parameters via rules is very comfortable and can be applied to existing components without restructuring parameters or rebuilding features. Since all rules are defined in context of the assembly itself, no data link to other documents like the skeleton exists.

Disadvantages of this approach are the lack of geometry transfer and the transferability of the iLogic rules. If in the example above the beams' cross section also should be modified via iLogic, eight more rules for manipulating the sketch geometry are necessary. Replacing a component within the assembly leads not only to re-constraining but also to adapting the corresponding rule to the new component since name and possibly parameter names have changed.

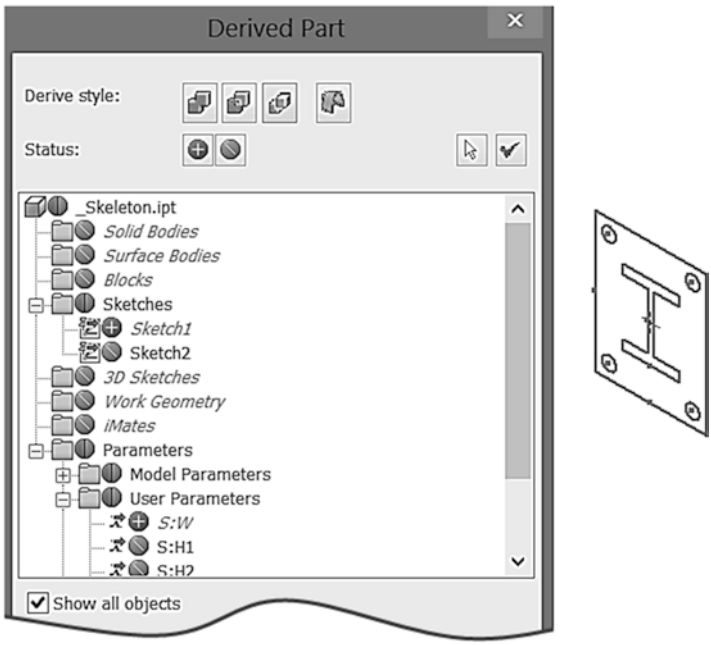


**Fig. 17.8** Derived geometry and parameters for the beam model

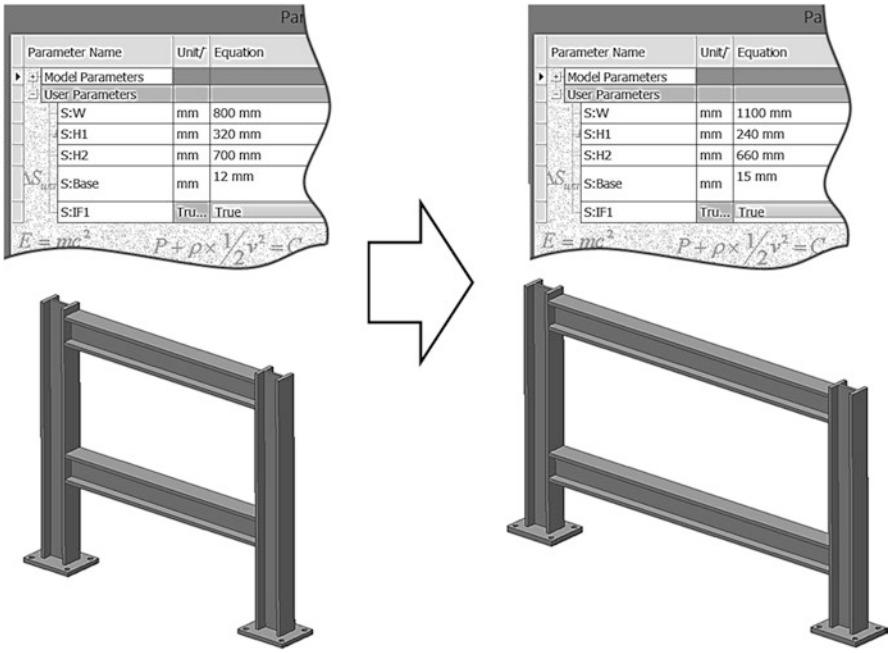


**Fig. 17.9** Skeleton controlled assembly document

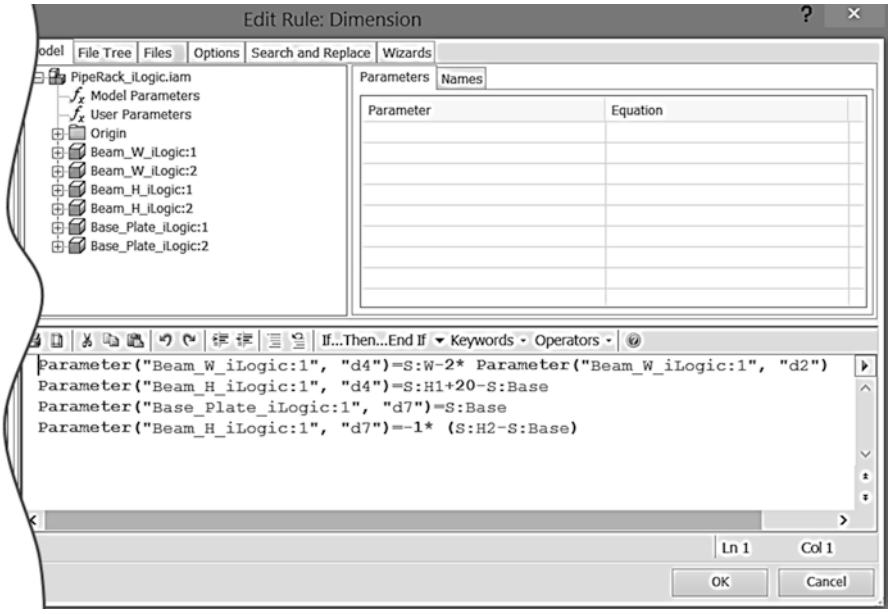


**Fig. 17.10** iLogic rules for the configuration of the base frame

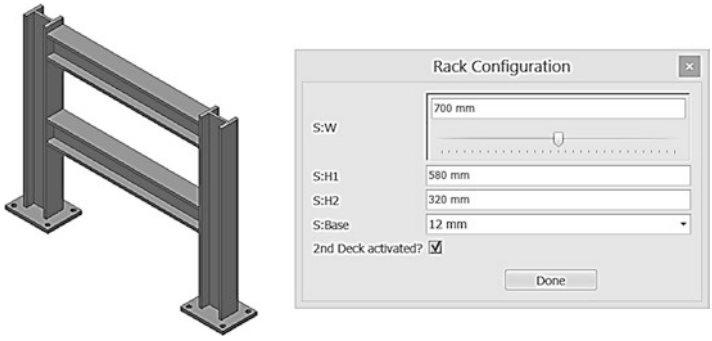


**Fig. 17.11** iLogic form

Both of the assemblies mentioned previously can be used as intelligent templates or assembly building blocks. To choose which approach is the best for modeling or if an intermediate approach has to be used depends on the specific modeling task, the degrees of freedom regarding parts as well as topology and the manifold of the solution space. A detailed investigation is beyond of the scope of this paper.

| Name | Value | Unit | Comment |
|---|---|---|---|
| S:W | 700 | mm | Width |
| S:H1 | 620 | mm | Height Deck 1 |
| S:H2 | 300 | mm | Height Deck 2 |
| S:Base | 12 mm | | Thickness Base Plate |

**Fig. 17.12** Inventor embedded Excel table

### *17.3.4 Spreadsheet Driven Design and Design Configuration*

As discussed in Sect. 17.2.2 an externalized parameter management within a spreadsheet application offers the possibilities to use extended mathematical and statistical operations. In order to use MS Excel for parameter processing it is necessary to define the parameters correctly. Since the format of an Inventor parameter is strictly defined the spreadsheet has to be built up as depicted in Fig. 17.12 with name, value, unit, and comment. Additional columns can also be used for user interaction, pictures, etc. but these will be ignored by Inventor.

When the spreadsheet is set-up it can be imported in Inventor in the parameter table dialog. The implementation can be done in two different methods. On the one hand linking the spreadsheet is possible. On the other hand the spreadsheet can be embedded in the Inventor file. The latter is favorable since the data link can easily be broken by moving the Excel-file to another folder, etc. The start cell is another important option. In the example above the first row contains only column titles but no parameters. So the starting cell has to be *A2* since Inventor has to start data processing from here.

Afterwards the parameters are listed as embedded parameters and can be linked via equations or design rules.

The use of MS Excel is a powerful tool for creating engineering or design configurators. Since Inventor reads only the first four columns of the first worksheet all other cells may be used for parameter calculation, linking parameters of standards or user interaction, e.g., plausibility checks or interactive diagrams. Some of the possibilities are shown in the application examples in Sect. 17.4.

### *17.3.5 Intermediate Result*

Regarding the basic parameter types named in Sect. 17.2.1, geometric and physical parameters can be entered directly within the parameter table in Inventor. In addition, process and technological parameters may be used since they usually express boundaries for geometric and physical parameters such as minimum bend radius or a hardening depth. Topological parameters control the suppression state of features and components. Since the state can either be true or false, a user parameter of the type Boolean is suitable.

All methods of parameter control as discussed in Sect. 17.2.2 are available in Inventor. Equations can directly be entered at dimensioning or in the parameter table. The externalization of input and calculation is possible due to the implementation of MS Excel. Additionally, the use of design rules is implemented in the iLogic concept.

This has to be understood as prerequisite for generating dynamic assemblies or intelligent templates as introduced in Sect. 17.2.4. Nevertheless, depending on the modeling task and the solution space, which has to be mapped into the model, an appropriate modeling technique has to be chosen.

Implementing reasoning techniques as presented in Sect. 17.2.6 is partially realizable. While rule-based and model-based approaches can be set up by use of the corresponding parameter control functionalities, the application of a highly developed cased based reasoning is to date not reported.

## 17.4 Application Examples

In the following subsections three application examples are shown. Some of them represent a combination of the functionalities mentioned above in order to address different control strategies and input formats (Fig. 17.13).

### *17.4.1 Base Frame*

The above assembly is used in multiple manufacturing stations as base frame. It consists of various rectangular tubes, connections plates, mounting feet, and a front cover. On top of the frame, a mounting plate with the assembly system and tools for the particular station is installed.

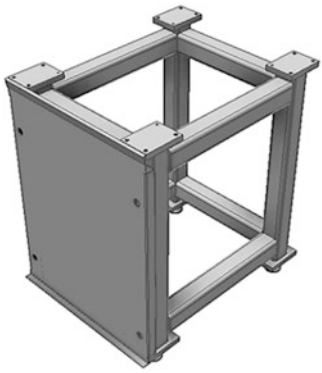

**Fig. 17.13** Intelligent template of a base frame

The base frame is modeled as template with iLogic rules for parameter calculation. In order to equip a new assembly station the frame is copied and the configuration parameters for height, width, depth, and clearance height are entered. If a change to the dimensions of the mounting plate occurs the assembly of the base frame can easily be adapted to the new boundaries. All drawings for the frame and its parts are updated automatically as well.

The design and documentation of a single frame was an 8 h task before automation. Every change to the mounting plate has led to the adaption of the single parts of the frame and took in average 1 h.

After automation the generation and maintenance can be done in minutes, the model is stable for all dimensions that usually occur.

### *17.4.2 Crank*

The crank depicted in Fig. 17.14 has to be used as dynamic assembly model since changes in the corresponding transmission design often occur. The assembly consists of the crank itself, a ball bearing and a retaining ring on the one shaft and two fitting keys on the other shaft which can be replaced completely by a spline shaft.

The assembly is modeled with a skeleton model behind the geometric one. In order to facilitate parameter input and definition, a spreadsheet is implemented in the skeleton model with a simple configuration user interface. As input, the geometric boundaries and type as well as size of the standard parts are defined. As output, the spreadsheet generates a report with strength calculation based upon entered loads. Additionally, iLogic rules have been defined to alter the size of the standard parts which have been set up as part families as well as the corresponding interfaces of the crank.

Design and documentation of the depicted assembly was done in 6 h before automation. Afterwards the configuration and reporting takes 10 min.

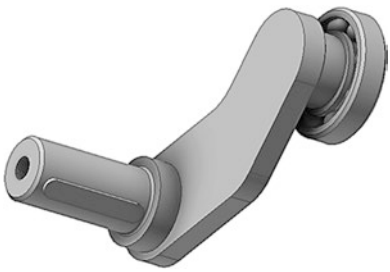

**Fig. 17.14** Dynamic assembly model of a crank

### *17.4.3 Platform*

Platform design is a common task in plant engineering. For quickly assembling a platform like illustrated in Fig. 17.15 a modular design kit was developed. It consists of building blocks for the base frame, stairs, connection bridges, and handrails.

Here also a combination of all of the above control strategies was implemented. The basic configuration is done via an Excel spreadsheet (Fig. 17.16) which calculates also most of the model parameters. Plausibility checks where defined in order to check whether the configuration is according to existing standards or has to be modified. The parameters are passed to a skeleton model for further computing and definition for the sketch geometries of beams and other parts. Within the main assemblies multiple iLogic rules are implemented for changing parts according to the configuration. Over 300 parameters and 138 part and assembly documents are managed.

The design of such a platform can be done in 40 working hours, after automation it is possible to assemble the above platform in less than 4 h. Some manual tasks remain since the layout plan has to be drafted and dimensioned for manufacturing.

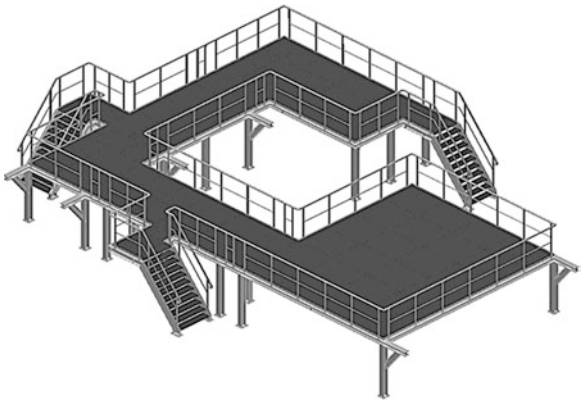

**Fig. 17.15** Modular design kit for platforms

| Name | Value | Unit | Comment | Part Name | Hint1 | Hint2 | | |
|---|---|---|---|---|---|---|---|---|
| M00:01 | 6000 | mm | Length of the platform in x | | OK! | if possible choose multiples of 1000 | size | |
| M00:02 | 9600 | mm | Width of the platform in y | | OK! | if possible choose multiples of 800 | | |
| M00:03 | 2800 | mm | heigth of the platform in z | | OK! | | | |
| M01:01 | 180 | | I-Profiles for Frame | | OK! | | beams and dimensions | |
| M01:02 | 200 | | I-Profiles for basement | | OK! | | | |
| M01:03 | 100x65x10 | | brackets | | | | | |
| M01:04 | 3 | oE | count of supports in x | | | | | |
| M01:05 | 4 | oE | count of supports in y | | | | | |
| M01:06 | 15 | mm | thickness of base plates | | | | | |
| M01:07 | 40 | mm | distance to support | | | | | |
| M01:08 | SP-30 - 50,8 - T 800 x 1000 | | grating | | OK! | | | |

**Fig. 17.16** Spreadsheet configurator

## 17.5 Conclusion

In the present paper different methods and functionalities for knowledge-based engineering have been introduced and discussed in a case study using the CAD system Autodesk Inventor Professional. We showed that building technical product configurators or engineering configurators within a CAD environment is possible.

Nevertheless, it is important to note that KBE is not suitable for all design tasks. Whenever the design is repetitive and can be expressed in explicit rules and so formulated as configuration problem, KBE is one of the best technologies for implementation.

There are still a lot of research questions to be answered. On the one hand we already pointed out at the application examples that choice and combination of different knowledge integration techniques and KBE functionalities is depending of the modeling task and the manifold of the solution space. Here guidelines and modeling principles as well as performance measures still have to be investigated. Another question is the implementation of reasoning techniques into the CAD models. Current research projects at our institute examine whether case-based reasoning might be implemented directly into a CAD configurator.

## References

1. Abramovici, M., Stekolschik, A.: Methodische Ansätze für die Weiterverwendung der digitalen Produktmodelle in der Produktentwicklung, Tagungsband 2. Gemeinsames Kolloquium Konstruktionstechnik, Verlag Saxoprint, Dresden (2004)
2. VDI: VDI 2218—Informationsverarbeitung in der Produktentwicklung—Feature-Technologie, Beuth Verlag, Berlin (2003)
3. Integrated Computer-Aided Design in Automotive Development, Springer Science & Business Media (2013)
4. Verhagen, W., et al.: A critical review of knowledge-based engineering: an identification of research challenges. Adv. Eng. Inform. **26**(1), 5–15 (2012)
5. La Rocca, G.: Knowledge based engineering—between AI and CAD. Review of a language based technology to support engineering design. Adv. Eng. Inform. **26**(2), 159–179 (2012)
6. Sabin, D., Weigel, R.: Product configuration frameworks—a survey. IEEE Intell. Syst. **13**(4), 42–49 (1998)
7. Hvam, L., Mortensen, N.H., Riis, J.: Product Customization. Springer Science & Business Media (2008)
8. Chapman, C.B., Pinfold, M.: The application of a knowledge based engineering approach to the rapid design and analysis of an automotive structure. Adv. Eng. Softw. **32**(12), 903–912 (2001)
9. Shah, J.J.: Designing with parametric cad: classification and comparison of construction techniques. Geometric Modelling, pp. 53–68. Springer, New York (2001)
10. Vajna, S.: CAx für Ingenieure: eine praxisbezogene Einführung. Springer Science & Business Media (2009)
11. Saufhoff, B., Lachmayer, R.: Generative design approach for modelling of large design spaces. In: Proceedings of the 7th World Conference on Mass Customization, Personalization, and Co-Creation (MCPC 2014), February 4th–7th, 2014. Lecture Notes in Production Engineering, pp. 241–251, Aalborg, Denmark (2014)

12. Peng, C.C., Rigdway, K.: Integration of CAD/CAM and spreadsheet data processing. Integ. Manuf. Syst. **4**(4), 29–36 (1993)
13. McDermott, J.: R1: a rule-based configurer of computer systems. Artif. Intell. **19**(1), 39–88 (1982)
14. Cox, J.J.: "Product Templates." A parametric approach to mass customization. In: CAD Tools and Algorithms for Product Design, pp. 3–15. Springer, Heidelberg (2000)
15. La Rocca, G., van Tooren, M.: Development of design and engineering engines to support multidisciplinary design and analysis of aircraft. Delft Science in Design–A Congress on Interdisciplinary Design. Delft University of Technology Faculty of Architecture Chair of Product Development Berlageweg 1 2628 CR Delft the Netherlands acjm eekhout@ bk. tudelft. nl/m. eekhout@ octatube. nl (2005)
16. Heinrich, M., Jüngst, E.W.: A resource-based paradigm for the configuring of technical systems from modular components. In: Proceedings of the Seventh IEEE Conference on IEEE, vol. 1 (1991)