Towards a Generic Framework of Engineering Design Automation for Creating Complex CAD Models

Gerald Frank, Doris Entner, Thorsten Prante, Vaheh Khachatouri
V-Research GmbH – Industrial Research and Development
Design Automation
Stadtstr. 33, 6850 Dornbirn, Austria
{Gerald.Frank, Doris.Entner, Thorsten.Prante, Vaheh.Khachatouri}@v-research.at

Martin Schwarz
Liebherr-Werk Nenzing GmbH
Technologiemanagement & Organisation Technik
Dr. Hans Liebherr Str. 1, 6710 Nenzing, Austria
Martin.Schwarz@liebherr.com

***Abstract*—For enterprises which engineer and produce highly customized products the reduction of design and manufacturing costs is of utmost importance. In this article, work towards a generic software framework for automating design processes resulting in complex customer-specific goods is described. Thereto, advanced product configurator user interfaces are tightly linked to CAD systems via an inference engine, which goes beyond product configuration in that it also facilitates generation of new parts. Knowledge-based engineering applications based on this framework support design engineers by automating those portions of a design process which are characterized by repetitive tasks. This is illustrated by two example use cases, namely design automation of ascent assemblies and box-type booms of cranes. On the application level, the implemented strategy is to reduce design-task complexity towards achieving significant speedups of up to 90 percent, which enables engineers to focus on creative, value-creating tasks. On the framework level, genericity and reusability is ensured by keeping the framework as CAD system independent as possible, by supporting different types of design procedures and complex assemblies, and by delivering added-value not only in the design and development phase but all along the process chain of integrated virtual product creation.**

Keywords—Engineering Design Automation; Knowledge-Based Engineering; Product Configuration and Generation; Software Framework; CAD.

I. Introduction

Production of industrial goods tailored to the requirements of ever smaller market segments and of individual customers has become a commonplace in many industries and is widely accepted as imperative to stay ahead of the competition when operating in highly developed markets. A direct consequence of that are higher complexities and smaller production batches, which can cause, in their turn, cost disadvantages. On the one hand, modular product design is widely advocated for combining the advantages of customization and flexibility with those of standardization and larger production batches. On the other hand, when it comes to products, where individualization/ customization entails complex engineering tasks to be performed per product and customer (termed engineer-to-order – ETO), other mechanisms of lowering costs while still maintaining the possibility of satisfying customers' specific needs have to be considered [1]. A powerful tool which is based on modularization and standardization, is the automation of (parts of) the engineering design process. This is generally termed *engineering design automation* and stands for numerous methodologies and applications in various industries, with the goal to automate the design process.

One approach to engineering design automation is knowledge-based engineering (KBE), which uses methodologies and technologies to capture and re-use knowledge of the product and its design process to reduce design and production time and costs. KBE can be seen "as a way of working intended to deliver engineering design

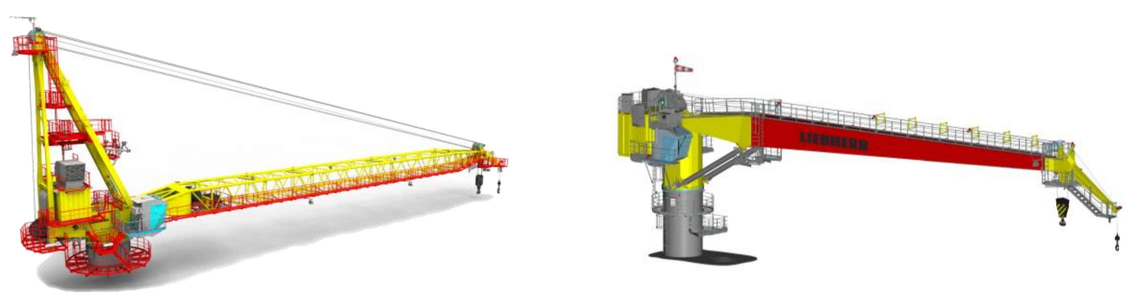

Figure 1. On the left, an example of a crane with attached ascent assemblies (highlighted in red); on the right, an example of a crane with a box-type boom (with the middle piece of the boom highlighted in red).

automation in scenarios where the retention of knowledge is critical" [2]. This includes the standardization of the parts/components of an artifact as well as the standardization and automation of the procedure which is used to assemble these parts.

Frank [1] presented a successful application of such an approach to ascent assemblies and box-type booms of cranes (Fig. 1). The application was realized in close collaboration with Liebherr-Werk Nenzing GmbH (LWN) [3], which is a manufacturer of a wide range of products including various types of cranes. The two mentioned use cases (ascent assemblies and box-type booms) are highly customized products, for which reason most orders involve the engineering department. Since this is a time intensive and thus very costly process, key aspects of the application are minimized design and production costs.

The KBE application is built upon a general framework, which uses a standardized part set as well as a rule base to represent the design knowledge. Furthermore, a structural analysis system is integrated in order to obtain the necessary statics requirements (for the use case of the box-type boom). The output of the application consists of a 3D CAD model as well as of the costs, bill of materials (BOM) and production drawings of the product.

This article further elaborates on the work presented by Frank [1]. First of all, the related work is more thoroughly reviewed and the approach of the paper is positioned clearly within the existing work. Furthermore, the underlying KBE system of the two mentioned use cases is discussed in more detail, giving an in-depth description of the steps from input to output as well as numerous illustrative examples. Finally, the general framework, on which these applications are based on, is also described in more detail.

The paper starts with a thorough overview of the related work concerning engineering design automation, including knowledge-based engineering and product configurators (Section II). After presenting the two mentioned use cases underlying our approach in Section III, a detailed explanation of the concept of the developed KBE application follows: In Section IV, an overview of the KBE system and its components (from input over the inference engine and the CAD model generation to the output) is given. The following Sections V to VIII discuss these building blocks of the KBE system in more detail. In Section IX, the underlying software framework is described. The paper closes with a demonstration of how the system is used in practice and its benefits for LWN (Section X), and a short conclusion and future work (Section XI).

II. Engineering Design Automation

The term of automated processes is most often associated with technological advancement in automation of production and manufacturing systems [4]. However, assuming the perspective of the overall product lifecycle, it becomes apparent that the highest potential of influencing costs lies in the earlier life phases of a product (Fig. 2; [5][6]). Namely, these are product and project planning (not shown in Fig. 2), design and development, and production planning, i.e., generating/preparing production-ready documentation such as the production drawings and means, and manufacturing details. In abstracting the design process as a spanning tree to show the transformation of the problem specification into a detailed technical description through a series of design decisions [7], the causality within the design process is clearly indicated, emphasizing the importance of the initial stages as they effect significant alterations in the steps that follow. Accordingly, the main focus of *engineering design automation* (as specified in more detail in the following paragraphs) is on generating added-value inventions and innovations towards automated design and product development processes.

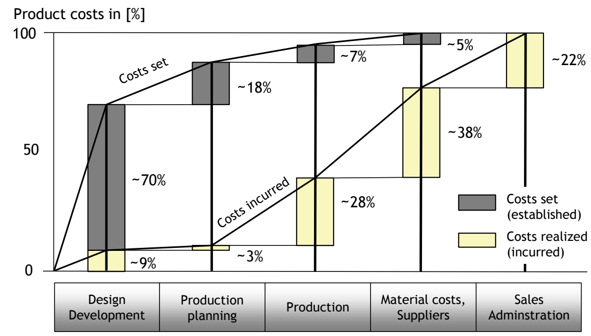

Figure 2. Determination of Production Costs.

Engineering design automation comprises a wide range of methodologies and applications within various industries, and dates back many decades. In the early 1970s, *electronic* design automation was the first commercially successful application, allowing the automatic design of circuits and electronic chips which already then became too complex for human engineers [8][9]. The first CAD and CAE (computer aided engineering) systems appeared as well in the early 1970s [6][10]. Nowadays, CAx systems (with x being a placeholder for, e.g., design, engineering, planning or manufacturing) are available for all stages within the product lifecycle. Ever since these first successful applications, engineering design automation has been implemented in many fields, such as the automotive industry [11][12][13][14], in aerospace and aircraft design [15][16][17][18], as well as in mechanical and plant engineering, as demonstrated in the use cases of the here presented work and of some earlier papers [1][19][20].

Generally, engineering design automation requires a deep insight in the design process to be able to capture and formalize the principles in the design domain. This again typically requires a set of building blocks (i.e., components/parts or modules), which can be combined in certain ways to result in the product (or part of a product/ sub-assembly) fulfilling the customer's requirements. Depending on the purpose of the automation task, the assembling procedure can be fixed (e.g., given by a set of rules) yielding exactly one solution, or capable of exploring various assembling strategies (e.g., have a stochastic component) resulting in a solution space [21]. Creating several solution alternatives is essential in the early, conceptual phase of the design process, whereas routine tasks are more prominent in the later, detailed design stage [22]. The approach of fixed assembling procedures is thus preferred for automating repetitive or routine design processes (consisting of nearly identical tasks), whereas methods capable of generating a set of possible solutions, by using different assembling strategies, are applied in innovative processes (including creative decisions; also termed computational design synthesis [21]).

In this paper, we concentrate on the problem of automating repetitive tasks, typical approaches for which are elaborated in the remainder of this section. We first review some literature on KBE and how it relates to expert systems and knowledge-based systems (KBS). Second, product configurators, which are one way of realizing a KBE application, are discussed. Finally, the approach of our system is outlined.

A. Knowledge-Based Engineering

The historical roots of KBE systems can be traced back to expert systems, which came up in the 1960s. The general term of an expert system is defined, according to Steinbichler [23], as a system that stores and accumulates specific knowledge of different areas and generates solutions in a user interface to given problems. Leondes [24] equates the terms "knowledge-based system" and "expert system". He also clarifies that a KBE system, the development of which is the topic of this paper, is a subset of a KBS.

In the field of KBE, methodologies and technologies are studied to capture and re-use knowledge of the product and the design process to reduce production time and costs, most often achieved through automating repetitive design processes [2]. The earliest ideas for KBE systems emerged in the late 1960s and early 1970s; more structured KBE systems have been around since the early 1980s [10][25].

According to Stokes [26], KBE can be defined as "the use of advanced software techniques to capture and re-use product and process knowledge in an integrated way." Thus, if applying the KBE approach, users' expertise has to be acquired and stored. A major advantage of this approach is that the captured knowledge is permanently available, and hence, the product development can be regarded as a holistic process. Thereto, all relevant design know-how is integrated into an overall product model, often stored in product data management (PDM) systems [27].

B. Product Configurators

In the context of automating repetitive processes, a common approach to the development of a KBE application is the use of a product configurator. *Product configurators* are defined in terms of finding a solution by combining components while fulfilling a set of constraints [28], or, putting it slightly differently, a configuration problem can also be understood as "the generation of a structure with predetermined properties by means of the combination of a certain number of objects" [29]. Bourke [30] expands this definition and describes a product configurator as "[...] software with logic capabilities to create, maintain, and use electronic product models that allow complete definition of all possible product options and variation combinations, with a minimum of data entries and maintenance". Another definition of product configurators is kept even more general, describing such a configurator as a tool assisting in the product design such that certain constraints are fulfilled [31]. In such definitions, parameterization of the components, i.e., dimensioning tasks such as adaptation of lengths and angles of a component, are considered to be part of the configurator as well. In [31], Brinkop presents a list of leading providers of product configurators in German-speaking markets.

Sabin et al. [32] classify product configurators according to their concept of configuration knowledge as rule-based, model-based and case-based product configurators. According to them, each approach represents the configuration knowledge and the instances of the product to be configured in a different way.

C. Our Approach

The KBE system approach presented in this paper not only uses complex product configurators, which allow for parameterization of the parts (e.g., adaptation of length or miter), but also incorporates the design of new parts (e.g., a handrail with arbitrarily angled corners formed from a basic straight part). This latter ability of the system to generate new parts goes beyond product configuration. The KBE system is based on an IT application of tested usability that supplies and processes knowledge and interacts with a CAD system.

The overall architecture represents a case of a customized system [33]. Keeping KBE functionality separate from CAD systems, while tightly interconnecting both via a bidirectional interface has a number of advantages. First, the captured and formalized design knowledge (in the form of parts and modular designs, and both simple and programmed rules as well as constraints) can be reused across different CAD systems. Second, also implemented and optimized variants of mechanisms processing and applying this knowledge remain available independent of the CAD system currently in use. The components guiding knowledge processing and application are, throughout this paper, collectively referred to as inference engine. For an overview, see the brown boxes at the center of Fig. 7 and the detailed discussions in Sections VI and IX.

Referring to Fig. 3, all kinds of input to the inference engine are also hosted within our KBE system/application and are thus kept CAD-system independent. This, third, gives us full flexibility to implement innovative user interfaces, without having to adhere to constraints set by design support for graphical user interfaces of any given CAD system. Beyond, fourth, the interface to structural analysis software is also managed by the KBE system.

Again referring to Fig. 3, the inference engine, at runtime, builds up a fully-fledged intermediate representation (i.e., an annotated tree structure, containing all necessary information about assemblies and assembly combinations), which can, fifth, be used to control different

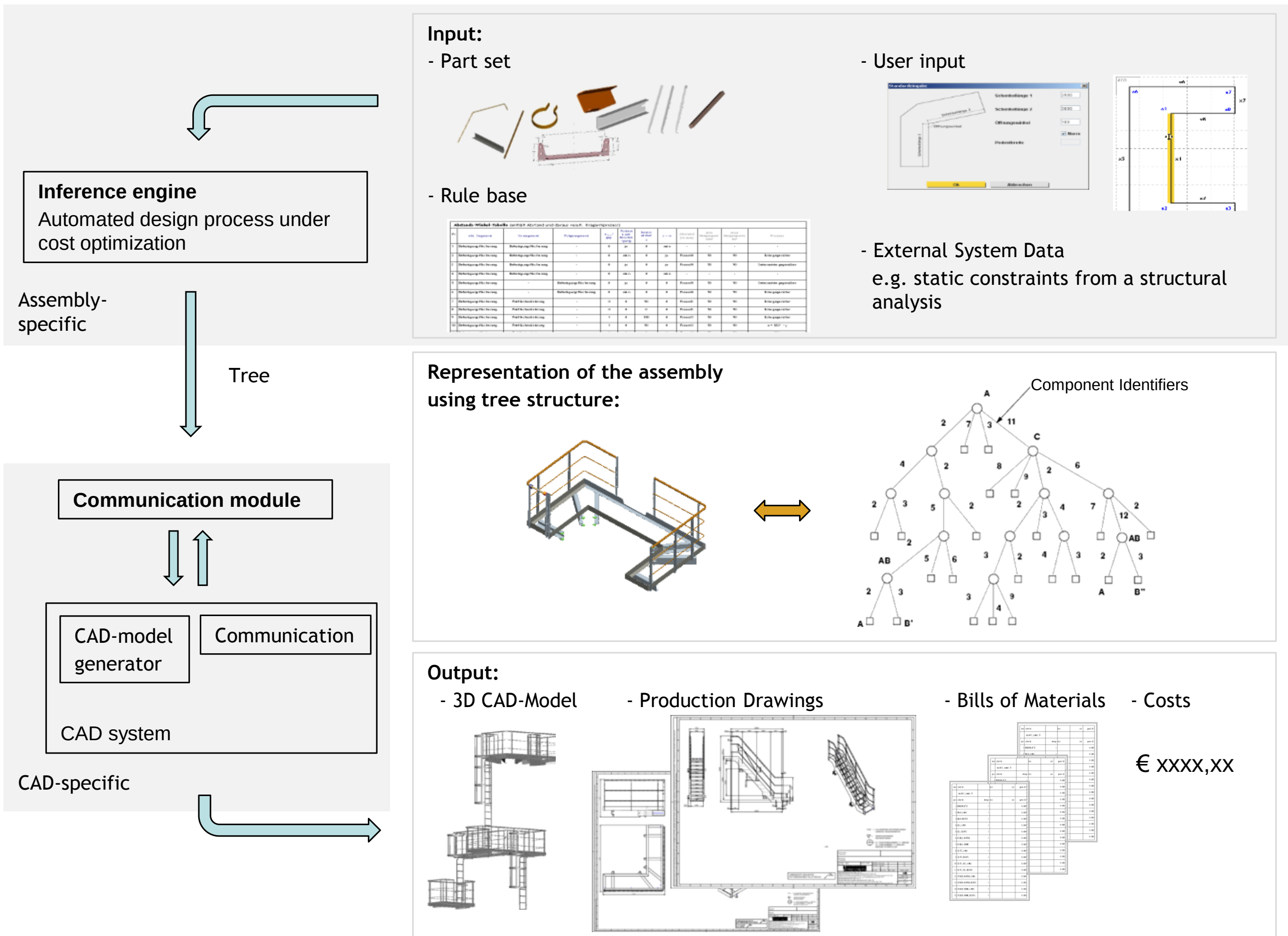

Figure 3. Overview of the KBE System, see Section IV for a detailed explanation.

CAD systems towards the generation of the customer-specific output shown in the lower part of Fig. 3. Here, the formation of CAD models and production drawings specifically necessitates the back channel to our KBE application, which is also indicated in Fig. 3. Finally, cost calculation is completely achieved within our KBE application.

Additionally, most of the just discussed functionalities and components are supported by generically implemented modules of our design-automation software framework discussed in section IX; exceptions to this are, e.g., our user-interface development environment and CAD-system application-programmer-interface (API). This, on top of the above presented advantages, allows for another level of reusability: Adaptations to other assembly types, differently tailored mechanisms of design-knowledge processing as well as combinations of such mechanisms.

In order to illustrate our approach and its advantages, applications to two use cases concerning the design of ascent assemblies and of box-type booms (see Fig. 1 and Section III for details) are based on the developed KBE system. The output of the KBE application comprises an automatically generated solution of the specified assembly, in accordance with a designer’s input and statics constraints, and its visualization as a 3D CAD model. As indicated in the discussion as to system architecture, our concrete KBE systems thus consist of a user interface, the inference engine, and a module to communicate with the CAD system (Section IV). The rule base contains all information about a product, i.e., its structure, function and behavior as well as its manufacturability and quality. In case an engineer designs the CAD model manually, this knowledge is all that s/he needs to fulfill the task.

Furthermore, the whole design process of a specific product is supported: Besides a 3D CAD model, production drawings, BOM and production costs are provided. That is, the complete engineering process is automated.

III. USE CASES

The development of a KBE application is demonstrated by means of two use cases in the field of crane design and manufacturing, as collaboration between LWN and the industrial research center V-Research.

LWN is a manufacturer of a wide range of products including ship-, offshore- and harbor mobile cranes as well as hydraulic duty cycle crawler cranes and lift cranes. Its mission is to fulfill customers' needs, and hence, to design and manufacture cranes according to the customers' requirements. While standardization is possible in many of their products, there are also segments which require customized adaptation of the crane to specific market demands (ETO). This can result in a partially or completely new engineered crane. In particular, the design of ascent assemblies for offshore, ship and harbor cranes as well as box-type booms for offshore and ship cranes requires substantial efforts. The costs related to the design of these assemblies are a major part of the overall engineering costs. Thus, we use these two types of assemblies as the use cases for our research for developing a KBE application to standardize and automate the engineering design task.

In Fig. 1, an example of each of the two use cases is given. In the left image, an offshore crane with attached ascent assemblies, highlighted in red, is shown. For maintenance, inspection and operating the crane, several strategic points on the crane, e.g., the machinery house, the pulley blocks, the winches and the operating cab, have to be easily accessible. The position of these access points depends on the crane design, which typically is customer specific. Therefore, an ascent concept, consisting of multiple ascent assemblies (platforms, roundplatforms, guardrails, stairs, or ladders), has to be developed anew for most orders. Each of these assemblies has a large range of variations in its specifications. For instance, platforms can have different shapes (rectangle, L-shape or U-shape, of almost any dimensions and angles), entries for ladders and stairs, and passages to connect to other platforms. The characteristics of the assemblies and their connections have to be specified by the designer according to the required access points.

The right image of Fig. 1 shows a ship crane with a box-type boom, the middle part of which is highlighted in red. The boom of such cranes has to be engineered to fulfill specific customer requirements, consisting of lifting capacity as well as of working and interference areas. These requirements are derived from the design of the ship or platform, on which the crane will be placed, and allow for little variation. Therefore, the boom section has to be designed individually for each application. This type of boom consists of a pivot, a middle and a head section. While the pivot and head sections are standard parts, the middle section, highlighted in red in the image and representing the second use case, requires custom engineering. Using a structural analysis (Subsection V.D), the thicknesses of the plates, which are assembled to form the boom of a pre-specified length, as well as the number of stiffeners and bulkheads to stabilize the boom have to be determined based on the customers' requirements (lifting capacity, working range, interference area).

The main goal of LWN was to reduce design and production costs by improving the design process in co-operation with the industrial research center V-Research. To achieve this goal, the possibilities of automating and optimizing the development phase were analyzed and, based on the results, a KBE application was developed. An overview of this application is given in the following section; the details follow in the subsequent sections.

IV. Overview of the KBE Application

Based on the explained background (Section II) and the requirements of LWN with regard to the two use cases (Section III), we developed a concept for automating the design process using a KBE system, including the integration of structural analysis. An overview of this approach is shown in Fig. 3, illustrated by the example of an ascent assembly. The details of the steps outlined in the figure are explained in the following sections. Here we just give a brief overview.

The *input to the KBE application* (Section V) consists of two types of information.

1. The first kind of input, consisting of the part set and the rule base, only depend on the assembly type (in Fig. 3 a platform), but not on the characteristics of the assembly (e.g., shape of a platform). The part set and the rule base result from formalizing and standardizing the engineering design knowledge (Subsections V.A and V.B). This type of input can be stored externally, e.g., in part templates or xml files.
2. The second type of input determines the characteristics of one specific artifact (e.g., the shape of the platform), and is obtained via interfaces. This information consists of the user input via the graphical user interface (GUI), e.g., the shape of the platform (Subsection V.C), and potentially some input from other external systems. For the use case of the platform, no such information is needed, however, for the box-type booms information about static constraints has to be retrieved from a structural analysis (Subsection V.D).

The input is then processed by an *inference engine* (Section VI), combining the information from the various input sources, as well as incorporating the logic for minimizing the costs. Both the input and the inference engine have to be adapted, based on the considered assembly (e.g., different rule base and user interface for a platform and a ladder). The inference engine includes procedures for generating new assemblies (Subsection VI.A), combining several assemblies (Subsection VI.B), and adapting already generated assemblies or combinations of assemblies (Subsection VI.C).

The output of the inference engine is an internal representation in from of a tree, representing the assembly and containing all necessary information: Every node of the tree represents a part of the assembly (or a subassembly, being a tree of parts itself) containing its geometry (e.g., length), its costs and its positioning information in reference to its parent part.

This internal tree representation is handed over to a CAD interface module to transform the information stored in the tree into data to generate the CAD model (Section VII). The communication module (and of course the CAD model generation) is specific to the chosen CAD system.

The output of the system is a 3D CAD model of the assembly, production drawings, BOM and the costs of the assembly (Section VIII).

The KBE application is based on a generic framework, which is explained in more detail in Section IX. As discussed in Section II.C, its core component, the inference engine, is designed such that only assembly specific tasks have to be changed. It is also possible to support different types of design procedures (e.g., bottom-up vs. top-down). Furthermore, the output in form of the internal tree representation, the generation of the CAD model, the calculation of the costs, creation of the BOM and the production drawings are as far as possible generically implemented. The communication module is also generic with regard to the assembly, and CAD system specificity is kept to a minimum, as the engine operates on a fully fledged intermediate tree representation for lossless bidirectional data exchange between framework and CAD systems. The thereby gained flexibility allows us to go beyond product configuration as its pure application is often too simplistic to cope with complexities encountered in today's engineering projects. As an example, the generation of new parts was discussed. Finally, on the other hand, graphical user interfaces are not yet implemented based on a framework or toolkit.

V. INPUT TO THE KBE APPLICATION

To model an engineering design process, it is necessary to investigate all factors that influence the design. The combination of these factors will lead to restrictions that have to be taken into account when formalizing the knowledge for the input of the KBE application. The main restrictions for engineering assemblies are the following:

- Industry and company standards,
- Statics requirements,
- Production costs,
- Implicit design restrictions (e.g., assembly erection or maintenance aspects), and
- Production restrictions (e.g., disposal factors).

For example, when designing a platform, the restrictions of the above bullet points result in platform entries conforming to standards (width, closing), or in special assembling logics, e.g., how to position the stays for the guardrail of a platform, how many pipes the guardrail requires and in which height they are positioned at the stays. Furthermore, the minimal and maximal realizable bending radius of a pipe is restricted by the manufacturing department.

A main challenge in building a KBE application is to extract the often implicit knowledge of the design engineers and formalize it appropriately. We next discuss this problem along the approach taken for both the ascent assemblies and the box-type booms.

A. Knowledge Acquisition

To build a KBE application, the relevant engineering expertise has to be acquired. As mentioned in Section II, engineering processes can be differentiated into repetitive and creative processes. In contrast to creative processes, repetitive ones consist of nearly identical tasks and are therefore independent of creative decisions. This condition is necessary for modeling the knowledge as a system of rules. In contrast to repetitive processes, creative ones occur typically only once. Because of that, modeling them as rules within reasonable time is economically not viable.

The main technique used to capture all relevant steps for designing the focused-on assemblies were interviews with the engineering experts at LWN [34]. In an iterative manner, interviews were conducted, the information was formalized, as well as the results were verified and additional questions were clarified in further interviews. The retrieved information served as a base for analyzing the repetitive design processes. Most of the time spent was used for identifying the restrictions of the bullet point list above.

Through the investigation of the design process of ascent assemblies and box-type booms, we found that the design process is indeed mainly based on repetitive tasks. One of the goals defined by LWN was that a specific repetitive design task should always result in the same, ideal solution. Because of the limited ability of a human to re-execute cognitive tasks identically, it is important to support users with a tool (i.e., a software application). Towards this end, the acquired knowledge has to be formalized, which is discussed next.

B. Knowledge Modelling – Part Set and Rule Base Design

The data obtained in the interviews first revealed that for the used building blocks of the assemblies a high amount of part variants existed (e.g., cantilever arms of arbitrary lengths). This, in turn, led to high costs not only in production but also in administration of the parts. To reduce the number of part variants, we developed a fixed set of standardized parts, which is sufficient for designing all required assemblies. This set of standardized parts contains not only fixed components (e.g., screws, stays, cantilever arms of discrete length), but also components which are capable of parameterization (e.g., adjusting to arbitrary length or miter of a component) and which may be generated completely new (e.g., a handrail with arbitrarily angled corners).

Based on the standardized part set, the knowledge about the design process was analyzed. Since the considered assembling processes are repetitive ones, designing these assemblies is based on a set of invariant rules that can be modeled and stored in an IT system. These rules represent directed dependencies in a form common for KBE systems, namely "IF (condition/-s) THEN (action/-s)"-statements, i.e., all conditions must be known and fulfilled before a rule can be applied [29].

Both the part set and the rule base are assembly specific. However, the approach of formalizing the knowledge in a rule set can be used for arbitrary types of assemblies. Rules can be changed without editing the source code. In addition, if a wide range of rules is acquired, nearly every form of assembly is supported. Therefore, repetitive tasks in

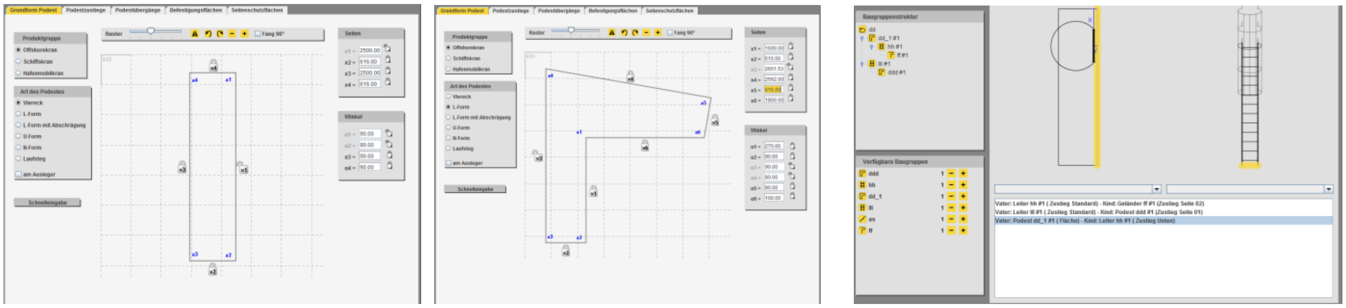

Figure 4. Graphical User Interface: Example of an interactive sketch for assembly dimensioning (left and middle image) and a wizard for the definition of an assembly-combination (right image).

designing new or adapting existing assemblies can be automated. This enables engineers to focus on creative, value-creating activities [35].

In the use case of ascent assemblies, an important subset of the rules belongs to the static requirements of the assembly. Each ascent assembly contains specific components (e.g., the cantilever arms for platforms, or stringers for stairs) which carry the main static load of the assembly and thus ensure the adequacy of the design with regard to safety and overall requirements of the structure. Based on these components, there is a limited set of parts with a fixed geometry (e.g., cantilever arms of discrete lengths). As a consequence, all statically relevant components can be pre-calculated using suitable software. The resulting parameters, e.g., the maximum load per square millimeter or the maximum gap to the next structurally relevant component, can be pre-assigned and therefore stored in rules. Based on the pre-calculated static parameters and the dimension of a given assembly variant, the number and/or dimensions of these parts are defined.

While the statics constraints of ascent assemblies can be represented by rules, the statics calculations of box-type booms are more complex. To verify static stability of these assemblies, dedicated structural analysis simulation algorithms have to be integrated into the design process (Subsection D).

C. *User Interface*

The GUI is an important component of the developed application. The focus was on minimizing user input, with the goal to allow users to define an assembly as efficiently as possible. The number and kind of input parameters depend on the specified assembly and the underlying rule set, such that design engineers only have to provide data which cannot be retrieved automatically. Furthermore, the user interface is supported by interactive sketches, and inputs are immediately visualized.

One tab of the user interface of the platform assembly is shown in Fig. 4. The interactive sketch in its initial form shows the starting-point platform layout. When changing, e.g., the shape and dimensions of the platform, this is immediately visualized, as shown in the middle image of the figure. Besides the geometry of the platform, various functionalities of the platform can be defined on further tabs of the user interface. These functionalities include for example platform entries and passages with arbitrary positioning (aligned at the left, right, or somewhere in the middle on the specified side of the platform), and several ways of closing (droplatch, gate, no closing). Which options are actually provided via controls in the GUI (e.g., if and how the geometry and functionalities of an assembly can be changed) depends on the specific type of assembly.

Every irregularity as to a defined process is highlighted by interaction dialogs. For example, if a design engineer defines inconsistent data, the application alerts the user.

In addition, a user is supported by some assisting tools. One of them is concerned with the combination of assemblies: a wizard visually supports the user to form a valid combination of assemblies (e.g., to define a complete access solution for an entire crane). The wizard shows all and only those combinations which are valid (e.g., it shows the possibility to attach a ladder or stairs to an entry of a platform, but not to connect stairs to a ladder). The right image of Fig. 4 shows the combination wizard for attaching the bottom of a ladder to the inside of a platform.

D. *Structural-Analysis Software Integration*

Structural analysis of an assembly is a major task in engineering design, in order to ensure the stability and safety of an artifact. In [36], the author defines structural analysis as follows: “Structural analysis is a process to analyze a structural system in order to predict the responses of the real structure under the excitation of expected loading and external environment during the service life of the structure.”

As explained in Subsection B, in the use case of ascent assemblies, the problem reduced to a few components of the ascent assembly, and all statics requirements could be pre-calculated and captured in a rule base. For the use case of box-type booms, the situation is much more complex, since all components of a boom are structurally relevant, and thus each individual box-type boom requires a separate structural analysis.

Because of the market-segment’s specific requirements and due to LWN’s commitment to fulfill customers’ demands (i.e., to provide arbitrary lengths and loads of the box-type booms), the dimensions of the boom vary considerably and cannot be limited to a standard set of parts. Thus, a structural pre-calculation of every possible

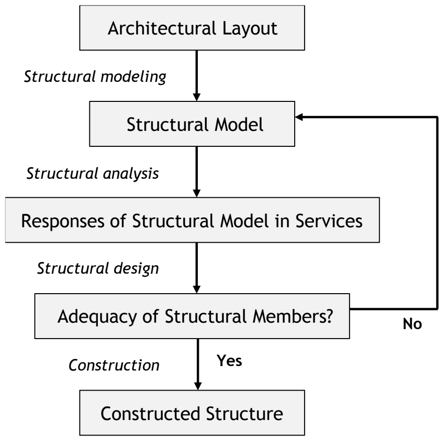

Figure 5. Structural analysis process.

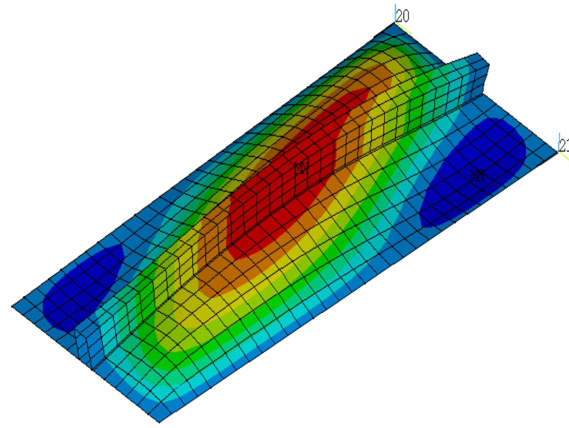

Figure 6. Result of a buckling analysis of a base plate section of a box-type boom.

dimension of the individual components and possible lifting capacities is not possible since the boom has to be considered as a complete system. Therefore, the logic regarding statics for box-type booms cannot be mapped to simple rules regarding its components. Hence, the developed KBE system integrates a structural analysis system (ANSYS [37]) to automatically obtain the necessary input regarding the statics requirements (e.g., thickness of the plates).

A standard structural-analysis process is shown in Fig. 5 [36]. This process applies to box-type booms at LWN in a similar way. In a first step, the design manager of the project converts the customer requirements into load cases. A load case mainly consists of a boom position (inclination angle) and a load capacity. After taking into account additional factors, a set of load cases is generated.

Based on this input, a simplified model is generated and processed by the structural analysis software ANSYS. In ANSYS, the model is analyzed with all the load cases. Based on multiple iterations, the defining parameters of the components (e.g., plate thickness) are optimized. As an example, Fig. 6 shows a visualization of the buckling analysis of a box-type boom base plate section. With the analysis it is determined how many stiffeners will be needed in a section to fulfill the requirements.

The output of ANSYS is an iteratively calculated optimal structure of a box-type boom: for each section, the material dimensions, part quantities and positions are defined. In particular, the calculated assembly structure consists of a weight-optimized geometry, i.e., the amount of used material is minimized. However, due to the nature of the boom production processes, a weight-optimized geometry is not necessarily cost-optimized, since production steps (e.g., welding) are not considered. Based on the results of the structural analysis, we developed an algorithm which uses a defined set of rules to translate the calculated geometry into a low-cost structure, while still adhering to the boundaries of the statics calculation. The parameters of the resulting structure (length and thickness of the plates) are used as input to the inference engine.

As the existing structural analysis procedure is a very time-consuming, complex and effort-intense process, its efficiency was increased by automating nearly all manual activities and integrating them into the developed KBE system. To supply the structural analysis software with all relevant information, a standardized data exchange format was developed. Now, the only manual activity consists of defining the load cases based on the customer's requirements. The developed method then transforms these data into an ANSYS-suitable configuration and hands them over to the structural-analysis simulation application. Once the simulation has started, no further user interaction is necessary. At the end of the simulation process, the structural engineer receives all the data for double-checking. For returning the results to the KBE system, an additional interface format was developed.

VI. INFERENCE ENGINE

The inference engine is the heart of the KBE application. It processes the input by combining the information of the standardized part set, the modeled rule base, the user input and potentially the input from the structural analysis. The methods and algorithms of the inference engine are developed such that the assembling of the single parts to the specified assembly happens in a cost-efficient way, i.e., an optimization logic (based on rules) is implicitly included. The content of the inference engine is assembly-specific; however, the structure follows a general framework (see Section IX).

The goal of the inference engine is to store all the information of an assembly in a framework-hosted general intermediate representation, which can be used to communicate this information to the CAD system. The chosen structure is a tree in which each node represents a part of the assembly (or a subassembly, being a tree of parts itself) containing its geometry (e.g., length), its costs and its positioning information in reference to its parent part. In the following subsections, we discuss the way the tree is built and the advantages of using such a structure when it comes to combining and adapting assemblies.

A. Generation of a Single Assembly

For the use case of box-type booms, only a single assembly (the box-type boom) has to be generated. In case of ascent assemblies, generating one single assembly (or several, not connected assemblies) is the simplest case. Typically, however, several assemblies (e.g., platform, ladder, stairs) are generated and combined, (see Subsection B for the combination).

For each assembly type (e.g., box-type boom, platform, ladder), one part is defined as the head-part, i.e., the root of the tree structure. According to the rule base and the design logic of the inference engine, all other parts are iteratively added to the tree as a child of an existing part. For example, the handrail is added as a child of the stay, from which the handrail starts. When determining the characteristics of a part, the following information is stored in the tree:

- The *geometry* of a part can be specified in two ways: (1) a part may be parameterized (e.g., by adjusting its length), or (2) a part may be generated from scratch by defining the required data (e.g., start, ending and corner points of a handrail, as well as an angle for each corner point for bending the handrail appropriately). For some parts, the geometry cannot be changed (e.g., bolts).
- The *positioning* information of each part contains the parent part in the tree (i.e., the part at which the current part is positioned in the CAD model), as well as the constraints how the current part is positioned in reference to its parents part (e.g., aligning two surfaces). The head-part of the tree is placed on a default position.
- Finally, the *costs* of each part, consisting of the material and production costs, are also stored in the tree.

B. Combination of Several Assemblies

As mentioned in the previous subsection, in the use case of ascent assemblies, typically several assemblies are generated and appropriately combined to form a complete ascent solution for a crane. For example, an ascent assembly may consist of a ladder, which connects to a platform, from which in turn stairs are mounted to reach another platform. As discussed in Subsection V.C and illustrated in Fig. 4 (right image), the definition of the combination is done via the GUI.

Each assembly (platform, ladder, etc.) is represented by a tree, as described in Subsection A above. To combine these assemblies to a single ascent solution, their trees have to be merged appropriately: For each such combination it is defined, how the two assemblies have to be connected, i.e., which part(s) of the first assembly is/are used as reference part(s) for which part(s) of the second assembly. Then, the trees of the assemblies can be combined accordingly into one bigger tree by simply adding the additionally required parent-child relationships between the corresponding parts of two assemblies.

C. Adaptation of Assemblies

For the use case of ascent assemblies, a further functionality of the inference engine is the adaptation of already generated CAD models of the KBE application (see Section VII for the generation of CAD models). If the engineer detects some inaccuracies or errors in the generated CAD model, the engineer could correct these manually in the CAD system, but also has the possibility to adapt the input in the GUI and re-generate the CAD model. In this case, not the complete model is re-generated, but only the necessary parts to reach the corrected solution are updated. This is done by comparing the trees of the initial and the updated model by traversing them systematically. In case of a detected difference in the two trees, the corresponding change is realized in the CAD model.

Such an adaptation could for example be necessary when a complex combination of ascent assemblies is mounted to a crane (in the CAD system), and intersections of the ascent assembly and the crane are detected. By, e.g., adapting the length of a platform, or replacing a ladder with stairs the mistake could be fixed (a detailed illustration is shown in Section X).

For the use case of ascent assemblies, the adaptation of assemblies or a combination of assemblies is of great importance since the generation of the CAD model is rather time intensive (many single parts have to be loaded and positioned in the CAD system). Using the adaptation generation only updated or new parts are loaded and positioned, saving a substantial amount of time. In case of the box-type boom, such an adaptation generation is not necessarily needed, since box-type booms consist of relatively few parts, and a major part of the calculation time is required for the statics calculations, not for the CAD model generation.

VII. CAD SYSTEM INTERFACE

Once an assembly is defined by the user, and, if necessary, the structural data are calculated, the respective data are handed over to the inference engine. After calculating all necessary information for generating a 3D CAD model and storing it in a tree structure, the computed data is sent to the CAD software in an iterative way (i.e., traversing the tree systematically). The communication between the inference engine and the CAD system is realized by using the API of a CAD system. The communication module encapsulates the handing over of the tree representation to CAD systems. This, being the only interface element, implements a clear and narrow interface as the basis for as much CAD system independence as possible.

When the data of a part in the tree is being sent to the CAD system, the part is first loaded and, if necessary, the geometry is adapted. Then, the part is positioned in reference to an existing part to ensure that all parts refer to each other. This is important since then every manual change directly

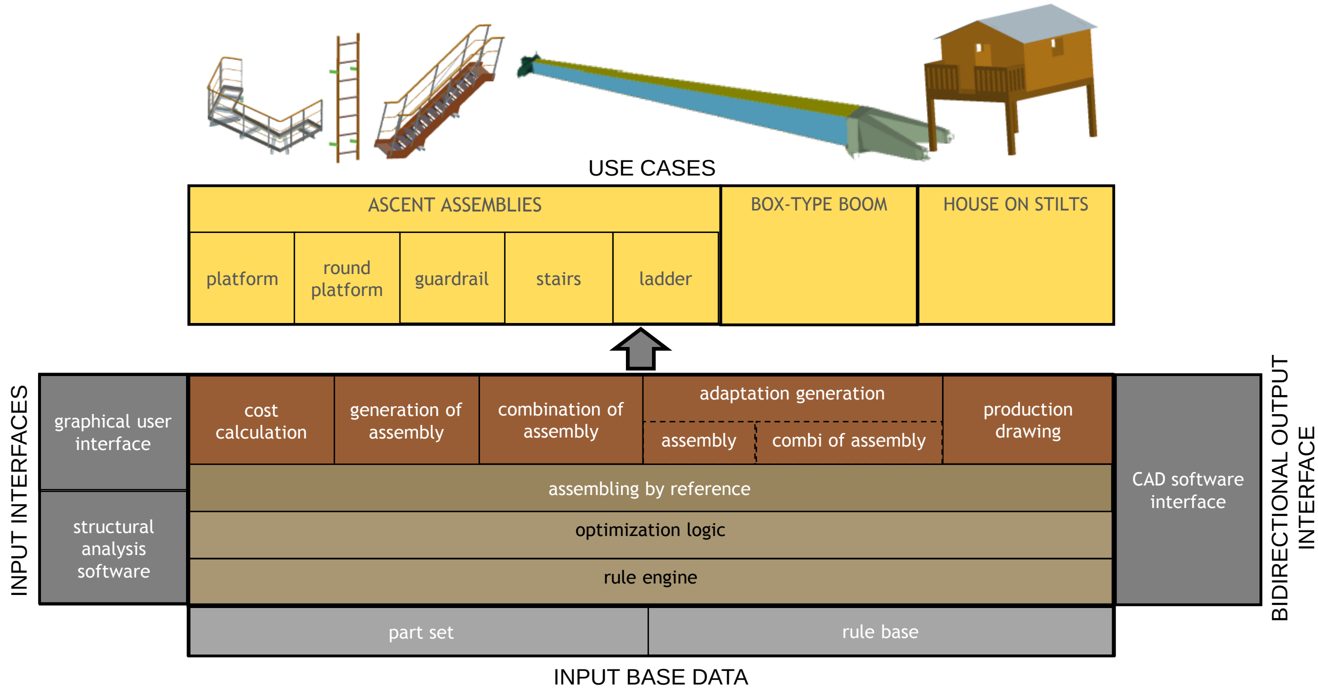

Figure 7. Overview of the framework, for details see Section IX.

affects all parts. For example, if a user manually changes the length of a part, the positions of all dependent parts are adjusted automatically.

The same principle has also been applied to assembly combinations as well as for the adaptation of assemblies. In the latter case, only adapted or new parts are loaded, changed (geometry) and positioned.

VIII. OUTPUF OF THE KBE APPLICATION

The KBE application outputs for the chosen assembly or combination of assemblies (in case of the use case of ascent assemblies) the following information:

- CAD model (see Section VII),
- Costs of the assembly/combination of assemblies,
- BOM, and
- Production drawings.

The costs and BOM are directly inferred from the tree generated by the inference engine. To complete the design process, production drawings have to be generated.

The positions of all required views of the product are calculated by an algorithm based on cut optimization, i.e., on efficiently using the space available on the drawing sheets. To ensure a good and fast solution, the concept of cut optimization was simplified. Each view is reduced to a rectangle or a combination of rectangles, which is/are put at the most appropriate and still available position.

After all views have been positioned, all production-relevant dimensions and the according measurements are, after determining the relevant dimensions in the inference engine, automatically added by our generic framework (Section IX). The framework is implemented with the API provided by the CAD system. It is based on a classification of dimension types.

Dimensions can relate to

- an edge,
- an edge to an edge,
- an edge to a point,
- a point to a point, or
- an angle.

For every mentioned type, a dedicated positioning function has been implemented. Finally, the BOM is added to the drawing.

IX. OVERVIEW OF DEVELOPED FRAMEWORK

The developed framework of the here described software consists of several modules. Fig. 7 visualizes all these components.

The brown boxes in the middle of the image represent the core of the software: the inference engine. The top dark ones consist of the main functionalities of the inference engine (generation, combination and adaption of assemblies) as well as generate the in Section VIII defined output documents. They use all or some of the functionalities provided by the underlying three modules “assembling by reference”, “optimization logic” and “rule engine” (which are also part of the inference engine).

For example, the modules for the generation of assemblies and their combinations use the optimization logic for calculating a tree representation corresponding to the assembly or assembly combination defined in the GUI. The optimization logic is based on a rule engine that retrieves all relevant rules and other data of the design knowledge (rule base), as well as adequate parts of the standardized part set (Section V.B). Thus, all necessary components, their geometry and their dependencies are calculated. The corresponding tree is established and completed with all necessary information. Then the module “assembling by reference” determines all information for positioning a component in the resulting 3D CAD model in reference to its parent part. Each component of the tree is enhanced by this information. In case of position changes, the advantages of assembling by reference become clear: The modification is handed over to all referenced components, and their positions are updated automatically.

The module “adaptation generation” is applied for the adaptation of already automatically generated CAD models (Subsection VI.C). This module uses functionalities for both calculating a new tree including the changes in the assembly specified by the user, and identifying differences between the new and existing tree.

The component “production drawings” also uses the described functionalities to automatically generate drawings as explained in Section VIII.

The task of the module “cost calculation” is the determination of the manufacturing costs. It uses both the existing tree and the rule engine to get all relevant data for calculation.

For data exchange, the core components of the inference engine use the interfaces the framework provides. There are two types of interfaces, which are prepared for assembly-independent use: Input and output ones (dark grey boxes). The input interfaces (currently GUI and structural analysis software) collect and determine all use case specific external data which are required by the inference engine to proceed with the calculation. The output interface enables the exchange of the tree, containing all necessary data about the assembly or assembly combination, between the inference engine and the specified CAD system to generate the mentioned resulting documents (Section VIII). Furthermore, it transfers all necessary CAD system specific information, particularly the identification number of a part, from the CAD system to the inference engine. This information is required by the inference engine to identify the parts in the CAD system. Thus, the established communication is bidirectional, i.e., from the inference engine to the CAD system and vice versa.

As illustrated in Fig. 7, in the context of projects with LWN, we used the developed framework for the design of the two above discussed use cases (ascent assemblies and box-type booms) as well as for houses on stilts (an illustrative case for an open-house day/event demonstration).

X. The KBE System in Practice and its Benefits

In this section, we demonstrate the practicability of the software and highlight the biggest advantages for the engineers of LWN when using the developed KBE system.

We first illustrate the design and functionality of the KBE system along the example of a platform. We focus here on the steps an engineer has to take to obtain a CAD model, costs, BOM and production drawings. Fig. 8 gives an overview of the general procedure.

As illustrated in the images in the top left box of Fig. 8, the engineer specifies the geometry (e.g., length and angles) and the functionalities (e.g., entry and fixing areas) of the platform. By pressing the “preview” button in the GUI, all data are submitted to the KBE system. In the background, the inference engine starts working: Using the standardized part set and the rule base, the developed algorithm calculates all parameters to generate a 3D CAD model and stores it in a tree. A connection to the CAD system is established and the data is transferred to the CAD system. The design engineer can now follow how each part is loaded and positioned in the CAD model, until the complete assembly is generated. The costs are automatically calculated and shown in the user interface (as illustrated in the left middle box of Fig. 8).

The design engineer then inspects the generated CAD model, and may detect some inaccuracies. An example is shown in the right box of Fig. 8: In Step 1 (current state), the engineer attached the platform to a crane (indicated by the yellow cube) and detected a gap between the crane and the platform. To correct the mistake s/he recalculates the necessary parameters to reach the target state, as illustrated in step 2. These parameters are then re-specified in the GUI (e.g., the length of one side of the platform). When pressing the button “change preview” in the GUI, the inference engine starts the calculation to adapt the previously generated CAD model and costs. By iterating these steps, the design engineer can easily and time-efficiently correct inaccuracies and mistakes in the 3D CAD model and reach a suitable solution to fit the customer’s requirements.

Finally, once the CAD model is ready, the engineer only has to press one more button (starting the required methods in the inference engine) to generate the production drawings and BOM (shown in the bottom left box of Fig. 8).

The software is not only adequately designed for its target groups, i.e., design engineers, but also offers flexibility in the design. This means that the standardized part set and the rule base together with the developed application enabled the standardization of the engineering process of nearly all ascent assemblies and box-type booms. In case the generation of an assembly would not be possible using the software, it still offers the possibility to automatically load the required parts of the standardized part set into the CAD system, where the design engineer can then manually generate the assembly.

The major benefit of the developed KBE applications lies in the time savings encountered in the engineering design process, and thus in lower costs and faster time-to-market. By automating the creation of new assemblies and the adaptation of existing ones, the complexity of design processes is well reduced and a significant speed-up is achieved. The engineering of ascent assemblies of an LWN offshore crane used to require up to 150 hours. Employing the here proposed software, these efforts can be cut down to 10 to 20 percent, i.e., to 15 to 30 hours. Also in case of the

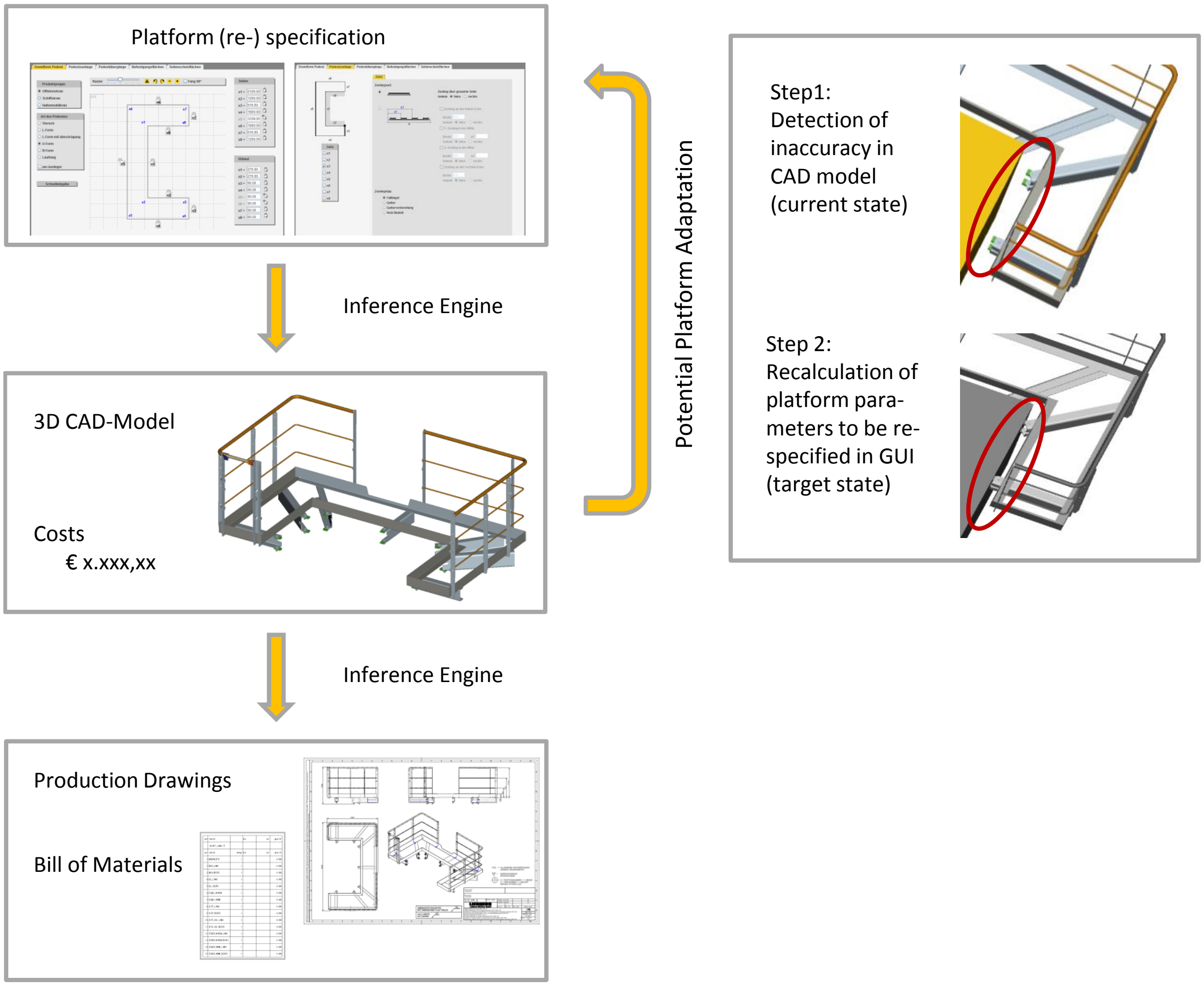

Figure 8. Automatic Platform Design Process, see Section X for details.

box-type booms, a significant speed-up in the development process could be observed, especially due to the integration of the structural analysis software into the KBE application.

XI. CONCLUSION AND FUTURE WORK

For companies operating in markets where highly customized products are predominant (and thus requiring a high variety of products, assemblies and parts), it is crucial to find means of reducing design and manufacturing costs. This paper presented a KBE approach to delivering engineering design automation tools, which help solving such challenges, and demonstrated its practicality and efficiency by means of two use cases.

The steps of the KBE system from input to output were thoroughly explained and illustrated with numerous small examples taken from the two use cases. From the beginning, the described application was developed towards a generic framework, which was also described in detail. While the framework was so far mainly used for the presented use cases of ascent assemblies and box-type booms, it is not limited to these tasks. An adaptation and extension towards other assemblies and components, as well as other fields (e.g., construction industry) is currently under way with promising first results.

Consequently, the software framework is designed as a flexible infrastructure towards realizing projects similar to the presented use cases also for other types of complex assemblies and in other industry sectors. Furthermore it is prepared to not only support variant but also adaptive design methodologies and to deliver added-value not only in the design and development phase but all along the process chain of integrated virtual product creation. Overall, a focus was put on keeping the framework as CAD-system independent as possible.

The software framework supports the development of advanced product configurator user interfaces, which are tightly linked to CAD systems via the core component of the framework, i.e., the inference engine. This engine operates on a fully fledged intermediate tree representation for lossless bidirectional data exchange between framework and CAD systems. Generally, bottom-up and top-down design procedures can be facilitated. This flexibility allows us to go beyond the boundaries of product configuration as the pure application of this paradigm is often too simplistic to cope with the complexities in today's engineering projects. As an example, the generation of new parts was discussed.

The shown user interfaces employ innovative concepts of user guidance supported by interactive sketches and wizards. Modular designs (i.e., standardized components) and both simple and programmed rules as well as constraints are used to formalize the critical knowledge, then automatically ensuring satisfaction of industry-wide and company-internal norms. A diversity of differently complex CAD templates can thereto be imported and used in the implemented approach.

If the design process of an assembly is based on a repetitive logic, it is possible to automate its generation. Furthermore, if a part set exists, which contains all necessary parts of an assembly type, and if the design know-how can be modeled in a rule system, an automatic design from scratch is possible.

The main challenge is to identify these repetitive design processes as well as determine and capture the engineering knowledge hidden behind these processes. By way of operational use of the presented methodology and KBE application in its engineering department, LWN gained valuable insight in the automation of engineering design processes, and further builds on this experience in future application areas.

The main benefit of the KBE applications is the significantly faster realization of the design process. For certain assemblies, this speed-up saved up to 90 percent of the design time. These savings in terms of time and thus also cost were realized with the presented application through the following features:

- Storage of the expert knowledge in a rule-base, together with a standardized part set,
- Reproducibility of all created assemblies,
- Enabling iterative engineering,
- Integration of structural analysis, and
- Production-suitable CAD models (i.e., models, characterized by feasible dimensions, tolerances and adequate material attributes for manufacturing them [38]).

Overall, through the automation of the repetitive part of the design processes of ascent assemblies and box-type booms the engineers of LWN nowadays save a significant amount of design time. This time can be used for creative, value-creating activities instead, such as preparing several design variants for customer specific requirements (e.g., ascent layouts) when bidding for an offer. Furthermore, the faster development of assemblies also allows LWN to react faster to changes in the market, as well as in the customers' requirements. Both of these aspects yield a competitive advantage for LWN.

XII. Acknowledgements

This paper discussed results and findings of a research project within the K-Project "Integrated Decision Support Systems for Industrial Processes (ProDSS)", financed through the Austrian funding program COMET (COMpetence centers for Excellent Technologies).

XIII. References

[1] G. Frank, "An expert system for design-process automation in a CAD environment," in 8th International Conference on Systems (ICONS), 2013, pp. 179-184.

[2] W. J. C. Verhagen, P. Bermell-Garcia, R. E. C. van Dijk, and R. Curran, "A critical review of knowledge-based engineering: An identification of research challenges," Advanced Engineering Informatics, vol. 26, no. 1, pp. 5-15, Jan. 2012.

[3] Liebherr-Werk Nenzing GmbH. <http://www.liebherr.com/en-GB/35267.wfw> 2014.05.28.

[4] S. Y. Nof, Springer Handbook of Automation. Dordrecht; New York: Springer, 2009.

[5] K. Ehrlenspiel, A. Kiewert, and U. Lindemann, Cost-Efficient Design. Berlin; New York: Springer; ASME Press, 2007.

[6] Committee on Engineering Design Theory and Methodology, Commission on Engineering and Technical Systems, and National Research Council, Improving Engineering Design: Designing for Competitive Advantage. National Academies Press, 1991.

[7] M. M. Andreasen and L. Hein, Integrated Product Development. Bedford, UK: IFS (Publ.), 1987.

[8] D. Jansen, Ed., Handbuch der Electronic-Design-Automation (Handbook of Electronic Design Automation). München; Wien: Hanser, 2001.

[9] D. Macmillen, R. Camposano, D. Hill, and T. W. Williams, "An industrial view of electronic design automation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 19, no. 12, pp. 1428-1448, 2000.

[10] G. La Rocca, "Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design," Advanced Engineering Informatics, vol. 26, no. 2, pp. 159-179, Apr. 2012.

[11] C. B. Chapman and M. Pinfold, "The application of a knowledge based engineering approach to the rapid design and analysis of an automotive structure," Advances in Engineering Software, vol. 32, no. 12, pp. 903-912, Dec. 2001.

[12] M. N. Le, Y. S. Ong, S. Menzel, Y. Jin, and B. Sendhoff, "Evolution by adapting surrogates," Evolutionary Computation, vol. 21, no. 2, pp. 313-340, May 2013.

[13] Y. Lin, K. Shea, A. Johnson, J. Coultate, and J. Pears, "A method and software tool for automated gearbox synthesis," in 35th Design Automation Conference, Parts A and B, San Diego, California, 2009, vol. 5, pp. 111-121.

[14] A.-C. Zavoianu, G. Bramerdorfer, E. Lughofer, S. Silber, W. Amrhein, and E. P. Klement, "Hybridization of multi-objective evolutionary algorithms and artificial neural networks for optimizing the performance of electrical drives," Engineering Applications of Artificial Intelligence, vol. 28, no. 8, pp. 1787-1794, 2013.

[15] P. Bermell-Garcia, W. J. C. Verhagen, S. Astwood, K. Krishnamurthy, J. L. Johnson, D. Ruiz, G. Scott, and R. Curran, "A framework for management of knowledge-based engineering applications as software services: Enabling personalization and codification," Advanced Engineering Informatics, vol. 26, no. 2, pp. 219-230, Apr. 2012.

[16] G. La Rocca and M. J. L. van Tooren, “Knowledge-based engineering to support aircraft multidisciplinary design and optimization,” Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, vol. 224, no. 9, pp. 1041-1055, Sept. 2010.

[17] S. W. G. van der Elst, M. J. L. van Tooren, B. Vermeulen, C. L. Emberey, and N. R. Milton, “Application of a knowledge based design methodology to support fuselage panel design,” The Aeronautical Journal, vol. 114, no. 1159, pp. 589-597, 2010.

[18] G. G. Wang and S. Shan, “Review of metamodeling techniques in support of engineering design Optimization,” Journal of Mechanical Design, vol. 129, no. 4, pp. 370-380, May 2006.

[19] G. Frank and C. Hillbrand, “Automatic support of standardization processes in design models,” in 16th International Conference on Intelligent Engineering Systems (INES), 2012, pp. 393-398.

[20] G. Frank, C. Hillbrand, and M. Schwarz, “Simulation based automated design to cost of structurally complex products,” in DS 70: Proceedings of DESIGN 2012, the 12th International Design Conference, Dubrovnik, Croatia, 2012, pp. 435-444.

[21] E. K. Antonsson and J. Cagan, Eds., Formal Engineering Design Synthesis. New York, NY, USA: Cambridge University Press, 2001.

[22] J. Clarkson and C. Eckert, Design Process Improvement: A Review of Current Practice. Springer London, 2010.

[23] G. Steinbichler, Methoden und Verfahren zur Optimierung der Bauteilentwicklung für die Spritzgießfertigung (Methods and procedures for optimizing the development of parts for injection-molding production processes). PhD Thesis, University Erlangen-Nuernberg, 2008.

[24] C. T. Leondes, Intelligent Knowledge-Based Systems Business and Technology in the New Millennium. London: Springer, 2004.

[25] C. van der Velden, C. Bil, and X. Xu, “Adaptable methodology for automation application development,” Advanced Engineering Informatics, vol. 26, no. 2, pp. 231-250, Apr. 2012.

[26] M. Stokes, Managing Engineering Knowledge: MOKA – Methodology for Knowledge Based Engineering Applications. London: Professional Engineering Pub., 2001.

[27] E.-S. S. Aziz and C. Chassapis, “A decision-making framework model for design and manufacturing of mechanical transmission system development,” Engineering with Computers, vol. 21, no. 2, pp. 164-176, Dec. 2005.

[28] R. Raffaeli, M. Mengoni, and M. Germani, “Improving the link between computer-assisted design and configuration tools for the design of mechanical products,” Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 27, no. 01, pp. 51-64, 2013.

[29] A. Brinkop, Variantenkonstruktion durch Auswertung der Abhängigkeiten zwischen den Konstruktionsbauteilen (Design of Variants by Analyzing the Dependencies among Parts Used in the Design). Sankt Augustin: Infix, 1999.

[30] R. Bourke, “Product configurators: Key enablers for mass customization.” Midrange Enterprise, Aug. 2000.

[31] A. Brinkop, “Marktführer Produktkonfiguration (Market leaders of product configuration),” Artikel, Brinkop Consulting, Mai 2013.

[32] D. Sabin and R. Weigel, “Product configuration frameworks – A survey,” IEEE Intelligent Systems and their Applications, vol. 13, no. 4, pp. 42-49, July 1998.

[33] S. Danjou, N. Lupa, and P. Koehler, “Approach for automated product modeling using knowledge-based design features,” Computer-Aided Design & Applications, vol. 5, no. 5, pp. 622-629, 2008.

[34] N. R. Milton, Knowledge Acquisition in Practice: A Step-by-Step Guide. London: Springer, 2007.

[35] H. Adickes, J. Arnoscht, A. Bong, R. Deger, S. Hieber, R. Krappinger, M. Lenders, P. Post, M. Rauhut, M. Rother, J. Schelling, G. Schuh, and J. Schulz, “Lean Innovation – Auf dem Weg zur Systematik (Lean innovation – Towards a systematic approach),” in AWK Aachener Werkzeugmaschinen Kolloquium ’08 - Aachener Perspektiven, Aachen, 2008.

[36] W. Kanok-Nukulchai, “Structural analysis,” in Civil-Engineering, K. Horikawa, Ed. Oxford: EOLSS Publishers Co Ltd, 2009.

[37] ANSYS. <http://www.ansys.com> 2014.05.28

[38] H. Brockmeyer, A. Lucko, and F. Mantwil, “Entwicklung von wissensbasierten Assistenzen zur frühzeitigen Produktbeeinflussung am Beispiel des Karosseriebaus (Development of knowledge-based wizards for timely influencing product properties, using the example of body making),” in Die digitale Produktentwicklung, 2008, p. 107 f.