Knowledge-based Engineering: An Efficient Method for Knowledge Processing

Y. Mvudi¹, J. H. C. Pretorius¹, L. Pretorius²
¹ Department of Electrical Engineering, University of Johannesburg
² Department of Mechanical Engineering, University of Pretoria

Abstract
Knowledge-Based Engineering (KBE) is a smart approach used in product development in order to shorten the duration of the engineering design phase. It consists of using computational intelligence to capture the design rules of a product family in order to generate several design variants from a single generative model. The KBE main objective is the automation of routine tasks that constitute an important part of the design phase. Such tasks can be performed several times as part of the design customization and optimization process. The use of KBE during the last three decades resulted in a significant reduction of the design duration in some companies. However, despite the impressive results obtained through the use of KBE in the automotive and aeronautical industry, there are still very few companies that make use of this approach. The review of relevant literature showed that the lack of an effective methodology of implementation is one of the major stumbling blocks to the expansion of KBE. Current methodologies do not seem to propose an efficient method for knowledge processing which is a very important phase of the implementation of the KBE approach.

This paper discusses a detailed qualitative method that addresses the issue of knowledge processing in KBE. Based on the system engineering approach and a logical classification of the design information, this method enables the efficient capture and presentation of design knowledge. The strength of this method lies in its ability to represent design knowledge in a form that makes it understandable to both engineers and programmers. This suitable representation has the potential to shorten the duration of the knowledge processing and facilitates the knowledge encoding phase. A practical example is also presented to illustrate implementation of the suggested method and to show its advantage over other methods.

Keywords
Knowledge-based Engineering, Knowledge Processing

1 INTRODUCTION
In this paper, an exploratory research method is followed to assess aspects of the traditional design engineering approach that consists of developing detailed technical model for each product under development. A close look at the field of product development reveals that very few products are developed from scratch. Radical innovations constitute only 20% of the product development activity whereas the remaining 80% consists of reusing and adjusting existing designs [1],[7]. This situation has led the product development industry to create efficient systems of design archiving and data management in order to facilitate the reuse of drawings. The development of CAD systems was a major breakthrough that fastened the design archiving and reuse. However, the process of generating an adequate design from an existing one can be tedious and time-consuming, especially for complex products. To address this issue, the CAD industry developed the intelligent modelling capabilities (parametric modelling) which allowed the automatic modification of a design through the variation of some key parameters [2].

Parametric modelling capabilities allow the capture of simple mathematical relationships that drive the design and facilitate the adequate adjustment of parameters to a new situation. However, in many cases the design knowledge may not be fully captured by simple mathematical relationships. Complex operations involving logical expressions and conditional statements are often required to represent the design knowledge. This is where Knowledge-Based Engineering comes in handy as indicated through the exploratory research presented in this paper. This new approach of engineering design makes possible the embodiment of complex design knowledge in CAD systems through the use of programming code.

During last decades, the use of the Knowledge-Based Engineering approach has led to impressive results in terms of the reduction the design phase duration. In the automotive and aeronautical industry, the literature mentioned up to 80% of the duration of the design time [1]. Despite the impressive results obtained through the use of knowledge-based engineering, there are currently very few companies that apply this approach. This is essentially a statement of the research problem
addressed in this paper. The lack of mature standard methodologies of implementation is mentioned in the literature as one the major obstacle to the diffusion of the KBE approach [1]. The development of a KBE application entails several phases and each of which has its own challenges. Knowledge processing is one of the phases of KBE implementation that requires a standard process. The process of capturing the knowledge and making it accessible to programmers is a real challenge. In this paper, a simple process of harvesting the design knowledge is proposed. Furthermore, a knowledge representation model is also suggested to facilitate the knowledge transfer between engineers and programmers. A simple example is also provided for illustration purpose.

2 CHALLENGES OF KNOWLEDGE PROCESSING

Knowledge processing is a crucial phase of the development of a KBE application. It consists of capturing the design knowledge of a product and presents it in a form that makes it programmable. The main challenge of this phase is the communication between engineers and programmers [3]. The lack of a common background between these two groups makes the communication challenging. Since the design knowledge flows from the engineer’s side to the programmer’s side, the onus is on the engineer to make the knowledge accessible to programmers. To date, very few solutions have been suggested to deal with this issue and few of proposed solutions have proven to be really effective.

Another important challenge in knowledge processing is the collection of the design knowledge embedded in engineers. Although design engineers know the rules needed to develop the product, it is extremely difficult for them to put this embedded knowledge on paper [3]. This challenge of documenting tacit knowledge is well-known in the field of knowledge management. It seems to be easier for engineers to use their knowledge than to document it. However, engineers are required to make this tacit knowledge explicit for the development of a KBE application.

3 CURRENT SOLUTIONS

There are two documented methodologies for KBE that have been found in the surveyed literature, namely: MOKA (Methodology and tools oriented to Knowledge based engineering Applications) and KNOMAD (Knowledge capture, Normalisation, Organisation, Modeling, Analysis and Delivery). Each of which proposes a particular approach for knowledge processing. It is important to be aware of these approaches in order to understand the current status of research in the field of knowledge processing for KBE applications.

In the MOKA methodology, there are two main tools used for knowledge processing: the knowledge classification and MOKA Modelling Language (MML) [4]. The knowledge classification standard stipulates that the engineering knowledge can be expressed in one of the five following forms: illustrations, constraints, activities, rules and entities (ICARE). The MML language is a representation of information in a form that makes it easy to be programmed in any programming language. The approach suggested for knowledge processing in the MOKA methodology consists of a two step process in which the engineering knowledge is first classified according to the ICARE standard and then written in the MML language. In this approach, the challenge of communication in knowledge processing remains because programmers have to understand the classified design knowledge in order to develop the MML representation.

The KNOMAD methodology suggested the development of an efficient ontology in order to classify logically and hierarchically the engineering design knowledge [5]. After this classification, the programmer can translate the design knowledge into programming code. Unlike the MOKA, there is no standard for the development of a specific ontology adjusted to the targeted product family. This is an area of concern because devising an ontology can be a tedious task for people who are not initiated in the field. Another issue with the KNOMAD approach for knowledge processing is the assumption that the programmer will be able to understand the ontological presentation and to translate it easily into programming code. The ontological representation of design knowledge can be very complex for someone that does not have the engineering background. Furthermore, there is no straightforward conversion process that allows the translation of the ontological representation into programming code.

Thus far, there seems to be a lack of proven method for knowledge processing, each KBE project develops a particular method adapted to the context. In this paper, a standardised approach of knowledge processing in KBE is suggested.

4 THE HIERARCHICAL APPROACH FOR KNOWLEDGE PROCESSING IN KBE

The hierarchical approach for knowledge processing is a methodology that aims at establishing a straightforward process for converting engineering design knowledge into programming code. This approach leverages the similarities between the system engineering approach and the object oriented programming (OOP). The main idea is to develop a process that can help engineers to capture the engineering
knowledge and to represent it in a form that makes it easily convertible in object-oriented language.

The hierarchical approach entails two main steps: the hierarchical representation of the model, the hierarchical representation of the design knowledge. These steps are detailed below.

4.1 Hierarchical representation of the model

The hierarchical representation of the model consists of developing a graphical hierarchy of the product based on the Work Breakdown Structure (WBS) used in system engineering. Skarka [7] acknowledges the importance of such hierarchy for knowledge acquisition. Using the WBS, a system may be split into several subsystems that interact together in order to achieve the overall mission of the product [8]. Subsystems may be grouped into a myriad of layers ordered hierarchically from the elementary components up to the whole system.

Since design engineers are very familiar with the system engineering approach, the development of a hierarchical representation of model comes almost naturally for a design team. By scrutinizing the physical architecture of the product, engineers can identify the main components of the system. The major subsystems are identified first and elementary entities are gradually identified through a top-down analysis [6]. In the hierarchical representation, each entity is represented by a circle. As illustrated in figure 2, each circle has to include the sub-circles representing the subsystems that it is made up of. This representation allows the clear and thorough identification of each components of the product.

Nomenclature is also an important part of the hierarchical representation. It allows the attribution of unique code to each subsystem (circle) identified. These unique codes play an important role in the second phase hierarchical approach. The code of each subsystem is a set of alphanumeric characters. The first character is a number that determines the level of the subsystem and the rest of characters designates hierarchically all the subsystems in which the subsystem is included. The level of the subsystem is given by the number of circles that enclosed the subsystem circle (including the subsystem circle itself). The remaining characters are a successive enumeration of all the letters that designate the circles enclosing the subsystem (including the subsystem itself).

4.2 Hierarchical Representation of the design knowledge

The hierarchical representation of the design knowledge is crucial step that addresses the communication issue between the design team and programmers [3]. It helps the design team members to represent the knowledge in a form that makes it accessible to programmers.

The hierarchical knowledge representation is based on a clear classification of the design knowledge. According to this classification, the design knowledge is made up of four main categories: requirements, characteristics, constraints and engineering rules. Requirements are specific data that are provided in customers’ requests specification or in the project scope. Requirements may also be derived by engineers from general pieces of information provided by customers or marketing department. It is important that requirements are presented as parameters that can only take precise set values or numerical values. Characteristics are a set of parameters that are necessary to manufacture a particular subsystem and/or to assess its compliance with the constraints. Yang and Redeisema refer to both characteristics and requirements as input data information [8].

Engineering rules are conditions and mathematical relationships that govern the model. Constraints are conditions that characteristics have to meet in order to have a valid model [4]. Constraints can be a restriction applied to one characteristic or a condition expressed as a mathematical formula involving more than one characteristic. Ratio and inequality are often used for constraints. On the other hands, engineering rules are conditional and logical expressions, and equations that help in determining the value of a characteristic based on the values of other parameters.

The hierarchical representation of knowledge consists of providing all the four categories of the design information for each subsystem (circle) at all levels. It is worth noting that each characteristic and requirement should be associated to a unique code in order to facilitate the writing of constraints and rules. While codification of requirements is left at the discretion of the engineer, the code for characteristics may be constructed by associating the code of the relevant subsystem to a particular character that symbolises the characteristic. This codification system implies that there must be a specific character that identifies each characteristic that belongs to the same subsystem.

4.3 Knowledge conversion table

Because of its similarities with the object oriented programming, the hierarchical representation allows an easy conversion of the design knowledge into programming code. Although the programmer does not understand engineering concepts involved in the process, he can easily find the equivalent code for each piece of information included in the hierarchical representation. Table 1 gives a good background for a straightforward conversion.
Hierarchical representation Knowledge | Object oriented programming equivalent
---|---
Entity (circles) | Object
Requirement | Variable
Characteristic | Attribute
Constraint | Conditional statement + mathematical expression
Engineering rule | Conditional statement + mathematical expression
Subsystem | Collection of objects

Table 1 - Knowledge conversion table

Through the use of this table, the programmer can easily develop the KBE application with any adequate CAD package.

5 CASE STUDY: AUTOMATED SHELF

The automated shelf is an intelligent device that enables the automatic storage, retrieval and collection of books. The purpose of this shelf is to automate the request, retrieval and collection of books. The next paragraphs describe how the hierarchical approach can be applied to the mechanical design of the intelligent shelf.

Using the WBS approach, the intelligent shelf may be split into three main layers and each layer is made of several subsystems that interact together. The circle diagram corresponding to the intelligent shelf is illustrated in figure 2.

Through the use of this table, the programmer can easily develop the KBE application with any adequate CAD package.

As indicated in the hierarchical representation theory (section 4.1), the first level is always the system itself. Based on the nomenclature rules and using the characters identifying each circle, the circles code will be as follows:

<table>
<thead>
<tr>
<th>Code</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The whole system</td>
</tr>
<tr>
<td>2a</td>
<td>The external cylinder</td>
</tr>
<tr>
<td>2b</td>
<td>The internal cylinder</td>
</tr>
<tr>
<td>3aa</td>
<td>The external cylinder lid</td>
</tr>
<tr>
<td>3ab</td>
<td>The external cylinder body</td>
</tr>
<tr>
<td>3ba</td>
<td>The internal cylinder axle</td>
</tr>
<tr>
<td>3bb</td>
<td>The internal cylinder slice</td>
</tr>
</tbody>
</table>

Table 2 - Subsystems Nomenclature

The second and final step of the hierarchical method is the representation of the design knowledge. This stage consists of stating the requirements for the whole system and providing characteristics, rules and constraints for each circle at each level. Lets take the example of the wing of the internal cylinder. The dimensions this part is strongly related to the size of the item to store in the shelf and the number of wings must be equal to the number of items. Firstly, requirement parameters are identified and encoded and then relationships between characteristics are expressed. The following tables illustrate the knowledge representation for this part.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Parameters (code)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item Size</td>
<td>Height (R3)</td>
</tr>
<tr>
<td></td>
<td>Surface (R4)</td>
</tr>
<tr>
<td></td>
<td>Weight (R5)</td>
</tr>
<tr>
<td>Item Quantity</td>
<td>Qty items (R6)</td>
</tr>
</tbody>
</table>

Table 3 - Shelf Requirements
These two tables illustrate how to express the knowledge required for determining the number and the dimensions of wings. Note that the characteristic “presence” allows the limitation of the number of wings by activating or deactivating the wings according to the number of items.

### 7 DISCUSSION

The hierarchical approach is an interesting method that has the potential to address current challenges of knowledge processing in KBE. However, a testing phase is required to establish the validity and the efficiency of this approach. It is important to apply this approach to a number KBE projects in order refine its steps and to assess its performance. This experimental phase is the only tool that can enable the hierarchical approach move from theoretical level to the practical one. Additionally, the experimental application of the hierarchical representation can help to evaluate its suitability to complex knowledge.

Another area of investigation is also the adjustment of the hierarchical representation to specific CAD packages. To date, there is a myriad of CADs with knowledge-based engineering capabilities and each package has a particular structure. It will be therefore interesting to establish straightforward conversion processes that enable a swift translation of the hierarchical representation to generative model for specific CAD packages. These conversion processes will facilitate the wide use of the hierarchical approach for knowledge processing.

### 8 CONCLUSION

The hierarchical approach suggested in this paper addresses the two main issues identified in knowledge processing, namely: knowledge elicitation for engineers and knowledge flow from engineers to programmers. The WBS and the design knowledge classification used for the hierarchical approach do not require intense training and it is easy to understand because engineers are very familiar with system engineering practices. The development of the hierarchical approach can help design engineers to document thoroughly the design knowledge without struggling to develop their own ontology. Another advantage of the hierarchical approach is the simplification of knowledge transfer between engineers and programmers. This is because the output of hierarchical approach is a knowledge representation that is easy to understand from a programming perspective and to convert in any language object-oriented with any adequate CAD package. Since the hierarchical approach has the potential to enhance the knowledge processing in KBE, an experimental study has to be undertaken as future research in order to assess the effectiveness of this method in practice.

### 9 REFERENCES


10 BIOGRAPHY

Yannick Mvudi holds a BEng in Electronic Engineering from University of Moulay Ismail. He has worked in the maintenance of electro-mechanical equipments and he is currently completing his Masters research project in Knowledge-Based Engineering at University of Johannesburg.

Jan-Harm Pretorius holds MIng and DIng degrees in Electrical and Electronics Engineering, respectively from University St Andrew in Scotland and University of Johannesburg. He was a technology manager of satellite applications centre at CSIR and he is currently professor at the Faculty of Electrical Engineering at University of Johannesburg.

Leon Pretorius holds a DEng in Mechanical Engineering and an MSc in Mathematics from University of Pretoria. He has more than 30 years of experience in professional and academic research and he is currently lecturer department of Mechanical Engineering at University of Pretoria.