COMPARING METHODS FOR IMPLEMENTING VBA IN CATIA FOR ELABORATING KBE PACKAGE

Wojciech SKARKA*, Łukasz ZAJĄC**

* Silesian University of Technology, Department of Fundamentals of Machinery Design
Konarskiego 18a, 44-100 Gliwice, Poland, e-mail: wojciech.skarka@polsl.pl
** Silesian University of Technology, Department of Fundamentals of Machinery Design
Konarskiego 18a, 44-100 Gliwice, Poland, e-mail: ukasz_zajac@poczta.onet.pl

Summary

The paper present comparison of methods for implementing VBA for creation of KBE application in CATIA system. The usage of VBA is one of the methods of automation of Generative Model creation in CATIA system. Creation of KBE Package forms one of the last phases of the whole process of KBE creation which consists of capturing knowledge necessary for KBE system, formal representation of knowledge, and transferring and implementing knowledge in KBE package. While creating KBE application, Generative Model is being formed and for the purpose of formalization of creation process of that model CATIA Knowledgeware tools are used. In order to have automation of that process it is necessary to ensure programming techniques, including VBA. The suggested 4 methods of implementing VBA for automation of Generative Model creation guarantee greater stability and repeatability of the model.

Keywords: Knowledge Based Engineering, Generative Model, CATIA.

PORÓWNANIE METOD PROGRAMOWYCH Z ZASTOSOWANIEM VBA
W SYSTEMIE CATIA NA ETAPIE TWORZENIE PAKIETU SYSTEMU KBE

Streszczenie

W artykule przedstawiono porównanie metod zastosowania VBA w końcowym etapie tworzenia aplikacji projektowej opartej na wiedzy (Knowledge Based Engineering - KBE) w systemie CATIA. Zastosowanie VBA jest jedną z metod automatyzacji tworzenia modelu autogenerującego w systemie CATIA, który jest głównym elementem aplikacji KBE. Tworzenie pakietu aplikacji KBE jest jednym z ostatnich etapów całego procesu tworzenia systemu KBE na który składają się akwizycja wiedzy koniecznej dla systemu KBE, formalna reprezentacja wiedzy i transfer i integracja wiedzy w aplikacji KBE. W fazie tworzenia aplikacji KBE tworzony jest model autogenerujący i dla formalizacji procesu tworzenia tego modelu wykorzystywane są narzędzia Knowledgeware systemu CATIA. Automatyzacja tego procesu wymaga zapewnienia technik programowych w tym zastosowania VBA. Zaproponowane 4 metody implementacji VBA do automatyzacji procesu tworzenia modelu autogenerującego pozwalają na zwiększenie stabilności i powtarzalności tego modelu.

Słowa kluczowe: Projektowanie oparte na wiedzy, model autogenerujący, CATIA.

INTRODUCTION

In designing processes of machines and devices computer aiding tools are commonly used and we can hardly imagine designing without using CAD, CAM or CAE tools. However, at the present stage of development it is not possible to talk about common usage of KBE (Knowledge Based Engineering) [2] systems in solving designing task. Two groups of such systems can be differentiated: the first one covers newly created systems whereas the latter one systems which are natural extension of already existing tools (e.g. CAD). Newly created KBE [11] systems, despite their positive effects of usage, are not commonly used. In the latter group the existence of modules in the existing CAD tools can be noticed, which enable knowledge implementation in these systems [3]. For the purpose of creating newly formed KBE systems, ordering methodology is usually used which arranges the process e.g. CommonKADS [4], MOKA [10]. In the process of development of KBE system the following phases are commonly differentiated [8], [10]:

- Identify,
- Justify,
- Capture,
- Formalize,
- Package,
- Activate.

The most important phases from the point of view of the creation of the very KBE tool are: *Capture*, *Formalize* and *Package* phase. In these phases the approach similar to suggested in MOKA methodologies, which are oriented to a given CAD tool can be used.

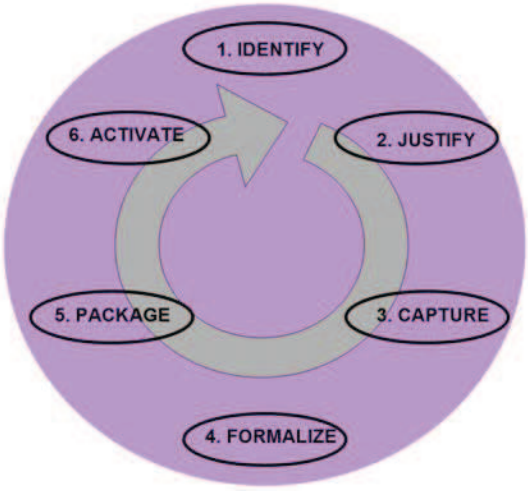

Fig. 1. General KBE lifecycle [8], [10]

For CATIA system such a methodology has been elaborated [8]. It is based on the model of MOKA knowledge base and uses Informal Model from that methodology in the process of knowledge capture. However, in that case MOKA metamodel has been adopted to use with CATIA system. It consists of two autonomous parts, with one being independent from the used CAD system and the other allows orienting to a given CAD system. In this way it is easy to adjust knowledge base which is being created to 'cooperation' with other CAD systems. During the process of Package, which is the creation of KBE tools, CATIA system tools are used for the formation of Generative Model [5], [6], [7]. This model can be created directly either by means of using Knowledgeware tools of CATIA system or by automation of the process by means of usage of programming tools e.g. VBA. It is one of the most time consuming tasks of that process and at the same time there is a lack of experience in that field. A big disadvantage is the choice of a given method of Generative Model creation by means of VBA, since there are at least a few hypothetical forms of that model to choose from.

Decisive factor which determines the choice is the level of time consumption as well as stability in using the model which results in proper generation of geometric model in the range of given sets of input data for designing process. In the further part of the paper four such methods have been suggested and compared which allow making *Package* process of KBE system creation easier with the aid of CATIA tools.

1. VB in VBScript

By means of using *Knowledge Advisor (KA)* modul in CATIA system it is possible to integrate *Visual Basic* code and CATIA *Part* file. *Visual Basic* code can be included in Macro as VBScript and in this case it can be used in *Power Copy* function. Script is activated by *Reaction* function that activates with the change of parameters (Fig. 2).

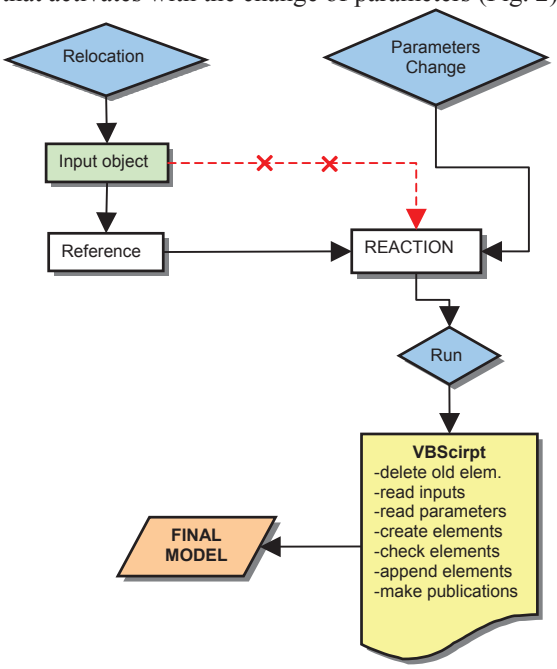

Fig. 2. VB in VBScript

Two CATIA *Open Bodies* have to be created before transformation. First *Open Body* should consist of references to input elements because it is necessary to have a set of unchanging names of references in VBScirpt. *Reactions* also have to be connected to those references. Connection of *reaction* directly with *inputs* cause errors while *inputs* are moved (red line crossed out in Fig. 2). Unfortunately, choosing such a kind of referencing makes it impossible to connect elements by *Formula* tool. Formulas are updated at the final stage as Power Copy is added and it generates errors. Table 4.1 shows examples how to create references to *inputs*. Second *Open Body* will be composed of products of VBScript.

Table 1: Example of references to *input* elements

Input Elements	References to input elements
Points	Points defined by coordinates
Directions	Lines defined by a point and a direction
Profiles	Parallel Curve with offset = 0

After transformation of basic elements (document, part, hybrid shapes etc.) VB code deletes all *Hybrid Shapes* in second *Open Body*.

Reading *input* elements and value of parameters should be made by searching method. The reason is that Power Copy adds some words to element names when it is activated. Next new elements are created with checking correctness of model. Only necessary features are appended to further work in CATIA tree. In conformity with the rules how to create KBE models, final features also have to be published.

Figure 3 shows example of profile created in this method. User through manipulating a series of parameters can get new model. VBScript checks that center point is in profile rage. To check location of center point virtual line between this point and COG (Center Of Gravity) is created. Then intersection of line and profile is made. If intersection exists (error = 0) than point will be out of rage and user is informed about that.

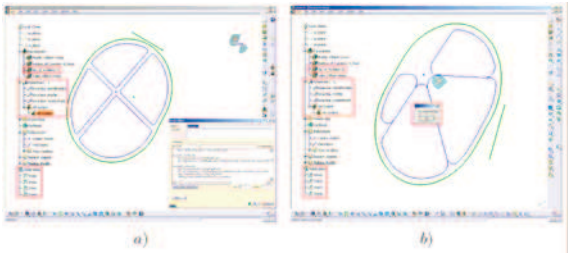

Fig. 3. Example of VB in VBScript usage

If one of the profiles is impracticable, user is informed about that and next profiles are created. To check correctness of profile *On Error Resume Next* function is used (see Fig. 3.b)

2. VBA as an environment to build complex model

This method is based on the usage of *SelecElemet2* function and it does not require the usage of *Power Copy*. Pointing at elements on CATIA tree allows creation of complete model in VBA. Interface is created in VBA, which allows user to point clearly at *input* elements and change value of parameters. Errors are identified by *check update error*. At first program checks if special *Open Body* exists, if it does not exist then *Open Body* will be created to include model, if it does exist then all Hybrid Shapes in it will be deleted. After that VBA program prepares to show window, values of parameters (as it was mentioned before) are read and a list of input elements is read. List of elements is created to enable showing what was pointed before and to enable *Cancel* option. Next user makes changes and decides whether to continue or to cancel work. If everything is satisfactory then a new model will be created or if *Cancel* option will be chosen then VBA will recall old parameters and create old model again. Figure 5 shows an example of this method. When user indicates tube profile which is not of circular shape he must also indicate a point from which place in a rage of profile one should start guide line. When it is impossible to create tube only guide line will be created to show user what is wrong.

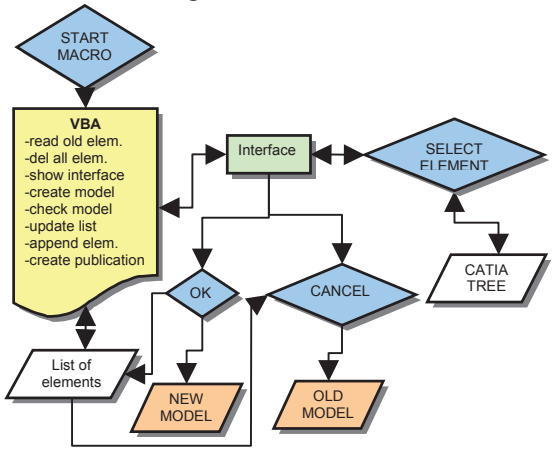

Fig. 4. VBA as an environment to build complex model

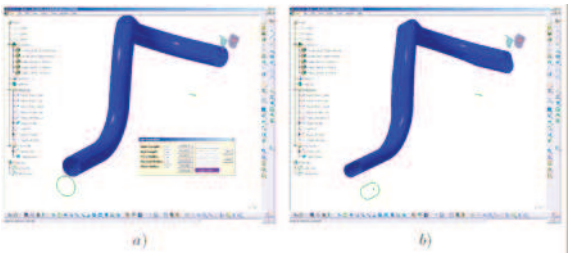

Fig. 5. Example of use of VBA environment to build complex model

3. VBA in conjunction with Power Copy

This method forms a combination of two previous methods. The interface is realized by VBA and management of model elements by *Power Copy* (Fig. 6). *Power Copy* has sets of correctly named parameters and *Open Bodies*. When *Power Copy* is inserted user points his input elements. In this case VBA does not have to read inputs when macro is started, and thanks to that interface becomes simpler. This method of transformation also includes creation of different design option in one VBA macro. Thanks to that a model has clear structure and is more stabile as compared with model created in CATIA (see Fig. 7). At the beginning VBA macro checks if *Power Copy* exists, if it does not exist then macro will stop and user will have to insert *Power Copy*. Next macro reads all parameter values and creates copy of it to allow canceling work and recalling old version. After that old model is deleted and a user has to make a choice of design option. When selection is accepted then a proper set of parameters is showed. User can change values of parameters and next macro checks if it is possible to create model. Finally model is appended on a CATIA tree. Flange model (Fig. 7) is an example

of implementation of this method. Macro can generate different type of flanges (stamped flange, stamped flange with chamfer and deep drawn flange). Each flange can have form of two to four screw holes. Additionally there is a possibility to generate flanges for tubes of other cross-section than circle. To sum up, macro allows creating twelve different types of flange.

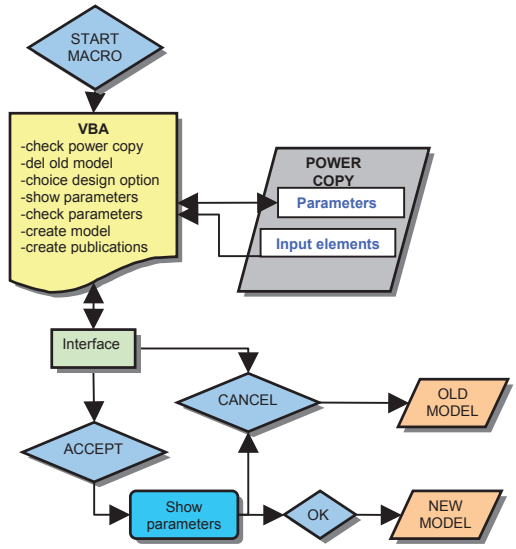

Fig. 6. VBA in conjunction with Power Copy

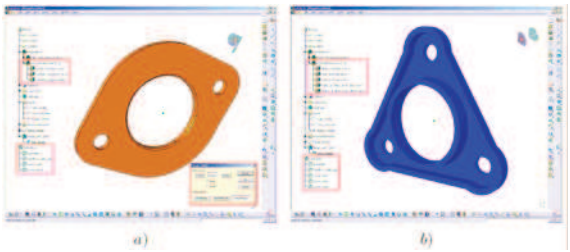

Fig. 7. Example of model build through Power Copy with VBA

4. PARSING ALGORITHM AND AUTOMATIC MODEL TRANSFORMATION

This method was elaborated to make automatic generation of VBA code from CATIA model possible. Basic idea was to use parsing algorithm to autotranslation models. First step of this idea is to read hierarchy of CATIA elements. This step is a tokenization process and it gives information about type of features of models. Every read element will be included in table with information about its parents. Reading the table by macro allows rebuilding a hierarchy of CATIA tree. Hierarchy must be read from bottom to the top then features that have no parents must be read first. While hierarchy is being read parser has information about type of each element and its parents and generates VB code to create this element.

The problem is how to build grammar as a piece of VB code. In CATIA model, especially in surface model, we have a lot of features and each has many creation options. Unfortunately every option of feature has different parents and so it is impossible to create all grammar by hand. In order to solve this problem it is necessary to use specialized libraries of CATIA system and application development software oriented to CATIA [1]. The above mentioned problems resulted in the fact that the method has been abandoned despite potentially promising possibilities.

5. SUMMARY

The paper presents comparison of programming methods including the usage of VBA at the phase of package of KBE creation. Three methods have been presented and carefully checked and verified, based on the examples of subassemblies of automotive industry. Additionally, the concept of the fourth method and its preliminary tests has been discussed. The presented methods have significant importance as far as automation of *Generative Model* is concerned and this model is the main object of the last three phases of methodology of KBE creation with the aid of CAD tools. These are the following phases: *capture, formalize* and *package*. The use of the existing CAD/CAM tool in the process causes considerable problems because it is necessary to maintain universality of the paradigm of conduct and, at the same time, its possibilities of specialization for the purpose of a given CAD/CAM tool [9]. The presented programming methods ensure this type of specialization for the purposes of CATIA system. All of the presented methods allow automation of *Generative Model* creation from the sources of knowledge gathered and recorded in the process of *capture* and *formalize* of the methodology. They constitute extension and automation in *Knowledgeware* function operation of CATIA system. So far independent usage of *Knowledgeware* function in practice has made it impossible for system creation of *Generative Model* [5, 6] due to the lack of automation aiding of its creation.

Table 2: Comparison of methods for implementing VBA for creation of KBE application in CATIA

Method	VB i VBScript	VBA as environment to build complex model	VBA with collaboration with Power Copy	Parsing algorithm and automatic model transformation
Pros	• Fast transformation • Control of model stability • Autogenerating of code by indicating element in CATIA tree (insert object resolution) • VBScirpt constitute an integral part of Power Copy	• Whole model is created in VBA • Stability control in VBA • It doesn't need Power Copy • User indicate input elements in CATIA	• Whole model is created in VBA • Stability control in VBA • Simply interface • Clear structure of model • Different design option in one VB macro	• Clear concept of the method • Universality • Easy possibility of development • Possibility of modularization
Cons	• Reaction causing errors when Power Copy is put and must be deactivated in Power Copy when it is created • Making references to input elements • Script Editor poorly supports writing code	• Interface complicated in use • Creation of interface takes much more time than translating model form CATIA to VBA	• Large VB code to include all design options • VBA must cooperate with Power Copy	• Problem to build grammar of CATIA model • Difficult to convert CATIA model to VBA • It has not been fully tested yet

REFERENCES

[1] *CAA RADE. IBM Software. Application Development (CAA RADE) Discipline.* www-306.ibm.com/software/applications/plm/caa/disciplines/caa/

[2] Hopgood A. A: *Intelligent Systems for Engineers and Scientists.* CRC Press LLC 2001

[3] Sandberg M.: *Knowledge Based Engineering – In Product Development. Technical Report.* Division of Computer Aided Design. Lulea University of Technology. Sweden, 2003

[4] Schreiber G. et al.: *Knowledge Engineering and Management. The CommonKADS Methodology.* MIT Press 2000

[5] Skarka W.: *Automation of designing process using generative models.* (in polish) Mechanik. 11/2005 s.781-782, Agencja Wydawnicza SIMP, Warszawa, 2005

[6] Skarka W.: *Developing Generative Models through Informal Knowledge Base.* Proceedings of the Symposium on Methods of Artificial Intelligence AI-METH 2005 and the Workshop on Knowledge Acquisition in Mechanical Engineering. p. 115-116 November 16-18, 2005, Gliwice, Poland, 2005

[7] Skarka W.: *Integrated Models as Knowledge Representattion in Designing Process.* (in polish) Mechanik. 1/2006 s.958-959, Agencja Wydawnicza SIMP, Warszawa, 2006

[8] Skarka W.: *Knowledge Acquisition for Generative Model Construction.* Concurrent Engineering CE2006, Antibes, France September 18-22, 2006 (accepted to be published)

[9] Skarka W., Mazurek A.: *CATIA. Modeling and designing.* Helion. Gliwice 2005

[10] Stokes M. (ed.): *Managing Engineering Knowledge; MOKA: Methodology for Knowledge Based Engineering Applications.* Professional Engineering Publishing, London, 2001.

[11] *The ICAD system* www.ds-kti.com/our_products/icad.shtml

Wojciech SKARKA, Ph. D. Eng. is an assistant professor at the Department of Fundamentals of Machinery Design. Since 1998 he was employed as a designer and a researcher at Mining Mechanization Centre in Gliwice. He is the constructor of many mining machines and the author of a dozen of patents. Currently he carries research on Knowledge Based Engineering and Interactive Electronic Technical Manual at Silesian University of Technology.

Łukasz ZAJĄC is a M. Sc student of Computer –Aided Design and Maintenance of Machinery at the Silesian University of Technology at Gliwice. He works as a research assistant on the KBE project in Tenneco Automotive – Henrich Gillet GmbH. His current focus is CATIA API - VBA environment and transformation strategy development.