IMPLEMENTATION OF AUTOMATION PROCESS IN GENERATING CAD MODEL

AZHAR BIN SHAFIE

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

IMPLEMENTATION OF AUTOMATION PROCESS IN GENERATING CAD MODEL

AZHAR BIN SHAFIE

This report is submitted in fulfillment of the requirement for the degree of Bachelor of Mechanical Engineering (Design and Innovation)

Faculty of Mechanical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

MAY 2017

APPROVAL

I hereby declare that I have read this project report and in my opinion this report is sufficient in terms of scope and quality for the award of the degree of Bachelor of Mechanical Engineering (Design and Innovation).

Signature :

Name of Supervisor : DR BASRI BIN ALI

Date : 12/7/2017

DECLARATION

I declare that this project report entitled "STUDY ON AUTOMATION OF GENERATING CAD MODEL" is the result of my own work except as cited in the references

Signature :

Name : AZHAR BIN SHAFIE

Date :

DEDICATION

To my beloved family, my future wife, supervisor and friends

ABSTRACT

The purpose of this study is to develop a program that allow the automatic generation of Computer Aided Drawing (CAD) model by implement concept of automation process in CAD software. Automation process has been widely used in industrial field that give positive feedback in terms of its convenience in the drawing process and time saving. An automation be able to help human on completing a job without the excessive work and energy required.

Automation process have a high reliability and more accurate in getting the best result in job instead of traditional drawing process and conventional use of CAD software. It also allow human to carries a multiple task in design creation for a shorter time. The program of automation will be develop using the application of Macros tools in CAD software. For this project, the CAD software that has been used to be an example for the case study is Computer Aided Three-dimensional Interactive Application (CATIA) software. The version used is CATIA V5 R21.

The system consist of two stage of developing process, which is the first part is the construction process for the CAD model and the second one is the coding development process. The model which for this project is a car rims is first created and recorded in CATIA through the recording tools in Macros without any mistake happened during the modeling process. Then the coding script of the recorded process can be called in the Macros tools to be edited and developed so that the automation process can work with the model.

The coding are developed by implying the usage of Visual Basic language in Visual Basic editor in CATIA. The important parameter coding line for creating the basic shape of the model are identified and modified in order to synchronize the model and the input system.

For this project, there are four main parameter that has been identified which is rim diameter, rim width, number of rim spoke and number of bolt hole. These parameter is required in the editing process when applying the application of dialogue box in VBA such as the user form. Original parameter is changed into the userform tools so that the input from the userform can be transferred into the model. The final model supposed to have an ability to alter into three different design with different parameter dimension.

ABSTRAK

Tujuan kajian ini dijalankan adalah untuk mencipta sebuah program yang boleh menjana sesebuah model lukisan secara automatik dengan menggunakan software Lukisan Berbantukan Komputer (CAD). Proses automasi telah digunakan secara meluas dalam bidang industri yang memberi maklum balas positif dari segi kemudahan dalam proses lukisan dan penjimatan masa. Proses automasi dapat membantu manusia untuk menyiapkan suatu pekerjaan tanpa urusan yang berlebihan dan tenaga yang diperlukan. Proses automasi mempunyai kebolehpercayaan yang tinggi dan lebih tepat dalam mendapatkan hasil yang terbaik dalam pekerjaan dan bukannya proses lukisan tradisional dan penggunaan konvensional perisian CAD. Ia juga membantu manusia untuk melaksanakan suatu tugasan yang pelbagai dalam proses penciptaan reka bentuk untuk jangka masa yang lebih pendek. Program automasi akan dibangunkan menggunakan aplikasi alatan Makro dalam perisian CAD. Untuk projek ini, perisian CAD yang telah digunakan untuk dijadikan sebagai contoh rujukan kajian adalah Permohonan Interaktif Perisian Tiga Dimensi Berbantukan Komputer (CATIA). Versi yang digunakan adalah CATIA V5 R21.

Sistem ini terdiri daripada dua peringkat dari segi pembangunan proses, di mana bahagian yang pertama adalah proses pembinaan untuk model CAD dan yang kedua adalah proses pembangunan kod program. Model rujukan dibina dan proses pembuatan model direkod di dalam CATIA melalui penggunaan alat rakaman dalam Macros tanpa melakukan sebarang kesilapan semasa proses pemodelan. Kemudian skrip kod program untuk proses pembuatan model yang telah dirakam boleh dikeluarkan semula di dalam alat Makro di mana ianya akan disunting dan dibangunkan supaya proses automasi boleh bekerjasama dengan model.

Proses pembangunan kod program dibangunkan dengan aplikasi penggunaan bahasa program Visual Visual yang sedia ada di dalam CATIA. Garis parameter proses pembangunan kod yang penting dalam proses pembinaan bentuk asas model perlu dikenalpasti dan disunting untuk menyelaraskan hubungan model dan sistem input. Untuk projek ini, terdapat empat parameter utama yang perlu dikenalpasti iaitu ukur lilit rim, lebar rim, bilangan jejari rim dan bilangan lubang bolt. Parameter-parameter ini perlu ada dalam proses penyuntingan semasa mengaplikasikan penggunaan *dialogue box* di dalam Visual Basic. Nilai parameter asal digantikan dengan arahan dari aplikasi *user form* supaya nilai input dari *user form* boleh dibaca oleh model lukisan. Hasil akhir model tersebut haruslah mempunyai keupayaan untuk mengubah bentuk ke dalam tiga reka bentuk yang berbeza serta ukuran parameter yang berbeza.

ACKNOWLEDGEMENT

I would like to express a highly special thanks to my project supervisor Dr Basri Bin Ali that continuously supporting and guiding me in knowledge and asset during this semester. A lot of knowledge that I received during last year in completing final year project and I indebted to who had made this final year project successful achieve the objective.

Also to mention, special thanks to my parents and family members who have supported me to complete this project in and encourage me in terms of spirits and financial assistance. And also thanks to my special love, Noor Farahana Binti Abdul Rahman who always support me in achieving the success of my coding process. Not to forget, thanks to all my friends and postgraduate students whom involved in this project. Last but not least, I would express my gratitude to the community in VB-Forum website who is willing to share with me their knowledge and give advice about my final year project. I also want to express my thanks to all person who involved in making this final year project possible to be done.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Definition
CAD	Computer Aided Design
CATIA	Computer Aided Three-dimensional Interactive Application
CAE	Computer Aided Engineering
CAM	Computer Aided Manufacturing
CTR	Click-Through Rate
GRIP	General Interactive Processor
SAGE	Semi-Automatic Ground Environment
2D	Two-Dimensional
3D	Three-Dimensional
MIT	Massachusetts Institute of Technology
PRONTO	Program for Numerical Tool Operation
PDM	Project Data Management
IBM	International Business Machine
VB	Visual Basic
VBA	Visual Basic Advance
VBScript	Visual Basic Script
GUI	Graphic User Interface
IDE	Integrated Device Electronic
DLL	Dynamic Linked Libraries

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Project designing is one of the vital process in every manufacturing industries. Computer Aided Design (CAD) is the use of computer systems to assist in the creation, optimization, modification, or analysis of a model for a design. The CAD modeling systems were developed to reduce time, cost and to minimize the trial and error adjustments.

A brief history of CAD shows that the first graphic system that was created was in middle year of 1950 by the US Air Force's SAGE (Semi-Automatic Ground Environment) air defense system. The system was developed at Massachusetts Institute of Technology's Lincoln Laboratory and involved the usage of Click-Through Rate (CTR) displays to show computer-processed radar data and other information.

Then in the year 1957, Dr. Patrick J. Hanratty which also known as "the Father of CAD/CAM" for his pioneering contributions to the field of computer-aided design and manufacturing, has developed the first commercial numerical-control programming system known as Program for Numerical Tool Operation (PRONTO). In year 1960 at Massachusetts Institute of Technology (MIT) Lincoln Laboratory, a person named Ivan Sutherland has produced a project called SKETCHPAD, which is considered as the first step to CAD industry.

In the past year CAD system softwares have their individual scripting languages for automation. Examples include AUTOLisp script language for AutoCAD and General Interactive Processor (GRIP) language for Unigraphics. But later with the use of Windows building blocks in CAD interface development, Visual Basic is used for scripting for all the modelling softwares such as Unigraphics, CATIA, SolidEdge, and SolidWorks.

Figure 2-1 Example of available cad software nowadays(Yip-Hoi, 2010)

The main purpose of the CAD is to create a prototypes to represent the construction of some system of interest while satisfying certain requirements of a user in an innovative way. As the CAD modelling techniques become more and more advanced, there is a need to complete the product modelling and design changes faster than ever. Updating the assemblies having hundreds of sub-assemblies and parts manually in Three-Dimensional (3D) modelling software is very complicated & time consuming.

The CAD models that have large number of parameters and high memory required more graphics and time to process the data. Hence updating the models affect the product lifecycle. A single error in the input data causes fatal error in the output results. To reduce the development time, minimizing the errors and introduce technologies faster to the market, many companies have been turning more and more to automation process.

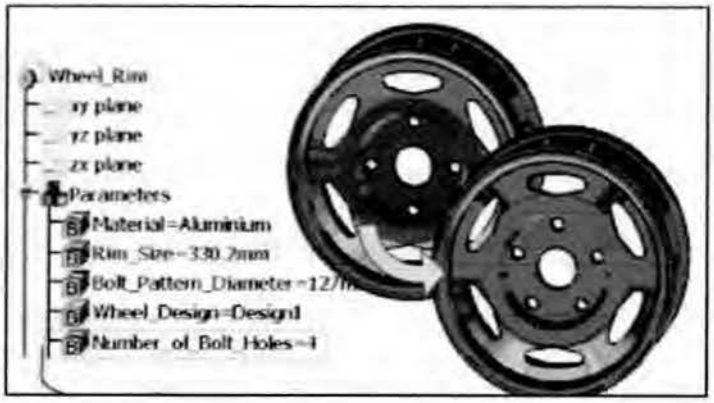


Figure 2-2 Example of product modification by automation

With the increasing demands and competition in industries nowadays, four important aspects are needed to be considered which are low cost, shorter lead time, improved product performance and the possibility to adapt products to different costumer specifications. One of the way to achieve these requirement is by adopting an approach where products are based on prepared design.(Chandrasegaran et al., 2013) By doing this, the design process can become more effective and efficient as if some of the work related to these products and design tasks are automated. The realization and implementation as well as its applicability of the automation method has given advantages to industry in terms of designing process as its allows for the shortened lead-time of product designs, cost reduction, more optimized product designs and customer tailoring, while giving the designers more time for creative problem solving.

Basically, the automation process in CATIA can be done with the usage of macros tools application. Macros use the Visual Basic language in the programing script for the coding. Through the application tools in Visual Basic programing such as dialogue box, interaction between the user and the program can be created without a problem.

1.2 PROBLEM STATEMENT

Instead of repeatedly creating the similar models or changing the parameters in the same models, engineers should be able to create program so that it automatically creates the new CAD models or to generate the further modification of the same model. This project shows the procedure for automation to generate the CAD models on the basis of coding by using the macros programming software in CATIA.

The automation process are meant to help in reducing the development time of a product especially in mass production industry. The process not only saves time, but also increases quality of data input for a model and minimizing the possibility of human errors. Thus, this criteria will lead to increasing in terms of productivity.

Commonly, most individuals have a different demand in the order of a product they wanted to have. This may causes a problem in the production process as different design are required to fulfill the different demand of the customer. So the automation are needed to solve this situation and ease the ordering process of a product. For an example, a shop that using the automation process can create a simple form from the application of CAD automation where the customer can order their product easily through online or personally at the shop. The customer are only needed to put in the parameter that they wanted in the order form and the design of the product will be automatically created in the CAD and the can see the model. This can be apply in many type of product such as the grill in the front car's bumper. As the customer need a special tools editor when they want to purchase their demanding product, this project can act as the platform to produce ordering method in more systematic way and also with the graphics display. Once the model has been selected, then the product can undergoes the manufacturing process such as rapid prototyping and adaptive manufacturing. This paper shows the macro programming method for generating the CAD models in CATIA.

1.3 OBJECTIVES

The main purpose of the work is to shift from manual modelling method to CAD automation process. The objective of this research are details as follows:

- To study the development of cad automation drawing.
- To develop the programming for standard part using CATIA programing.

1.4 SCOPE

This research project primarily focuses on how the automation work in CATIA V5 software. The project will include:

- A modelling of part design of three type of rim wheel with different pattern by using the CATIA software.
- Creating the script of coding by using visual basic language in macros tool.
- Executing and testing the coding.
- Make an adjustment and editing to the coding to fill the programming requirement

CHAPTER 2

LITERATURE REVIEW

2.1 GENERAL CAD DEFINITION

Computer Aided Design (CAD) is defined as a system that includes the use of information technology in the Design process. A CAD system consists of information technology hardware, specialized software (depending on the particular area of application) and peripherals. System software is the core in a CAD system, which makes use of graphics for product representation. For example, the usage of databases for storing the product model and drives the peripherals for product presentation. The application of CAD does not change the landscape of a process in design modeling but as stated in the definition of CAD itself, it aids the user to design a product.

The application of CAD initiated in the MIT from Ian Sutherland, when the first system which known as Sketchpad was created within the SAGE research project. At first, industry in automotive and aerospace field were the first users and the forerunners of development of CAD technology. The first system were very expensive as in those period of time the computer graphics technology was not so advanced and flourish yet.

The application and usage of CAD are listed as follows:

- Accurately generated and easily modifiable graphical representation of the product. The user can nearly view the actual product on screen, make any modifications to it, and present his/her ideas on screen without any prototype, especially during the early stages of the design process.
- Perform complex design analysis in short time.
- Implementing Finite Elements Analysis methods, the user can perform: Static, Dynamic and Natural Frequency analysis, Heat transfer analysis, Plastic analysis, Fluid flow analysis, Motion analysis, Tolerance analysis, Design optimization.
- Record and recall information with consistency and speed. In particular, the use of Product Data Management (PDM) systems can store the whole design and processing history of a certain product, for future reuse and upgrade. The technique initiated in the MIT from Ian Sutherland, when the first system.(Yip-Hoi, 2010)

2.2 CAD AUTOMATION

The automation can be defined into three categories which is design automation, manufacturing automation and administrative (or control) automation. For design automation, the process are related to the computer assisted drafting, design and engineering. The next one which is manufacturing automation is the use of various control systems for operating equipment such as machinery, processes in factories, and other applications and vehicles with minimal or reduced human intervention. Some processes have been completely automated by the computer controlled processes. Lastly for administrative automation is for computerized accounting, inventory control system and shop floor tracking system. (Ali & Ibrahim, n.d.)

Over the past 20 years, computer-aided design (CAD) software have become a mainstream media in most industry as a tools for project designing. As the time passing, the software has rapidly evolving and continuously improve to become a programs with enormous capabilities and function. (Lowe & Hartman, 2000) CAD software always been related with the automation since the beginning history of CAD as creating and editing a Two-Dimensional (2D) and 3D model is faster when using a computer compared to manual hand drawing. But despite how intuitive and advance the development of the existing CAD system, most user still desperately manually doing the same bland CAD translation and repair work. Because of that, the application of CAD automation can help to ease the operating process by the user.(Venkatesan & Karnan, 2009)

The concept of CAD automation enables the user to automatically created the design or adjusting the method and process of the already made design to fulfill the criteria of the new design. The process allows the user to automate translation and repair processes of the design that are already in large part by enabling users to translate multiple files at one time without having to open or view them graphically.(Ramly et al, 2013)

There are two method that commonly used for CAD automation which the first one is by manually writing a complete programing code and other one is the application of macro tool in the modeling software. Depending on the applications and the requirement for the desired model, a particular method that is suitable is used.(Weisenberger, n.d.)

2.3 CATIA

CATIA (Computer Aided Three Dimensional Interactive Application) is one of the modeling software that available for use. It is a multi-platform software commercially used for Computer Aided Design (CAD), Computer Aided Manufacturing (CAM) or Computer Aided Engineering (CAE) software suite and is a world leading software for those

application. CATIA is developed and owned by French company Dassault Systems and it is marketed worldwide by International Business Machine (IBM). (Wayzode, 2012)

Catia is the first generation for the graphic terminal. Initially, the primary objective and application of Catia was to develop a software to define the shape of an airplanes which began at Dassault Aviation in 1967. From years to year, the development of CATIA keeps to progress further into more applicable and intuitive software. (Bernard, 2003)

Basically, the structure creation of programming line for CAD software is differ depending on type of CAD software used. As for CATIA V5 program structure, revolve around four main criteria where it is start with the part body, then sketch plane, definition of the axis for the sketch and lastly apply additional feature to convert a two dimension sketch into a three dimensional object.(Wayzode, 2012)

CATIA offers the ability to write or editing the programing script of a model through the application of macro tool. A macro recorder in the macro tools help to ease the automation of CAD in CATIA software. The programmer, need to be familiar with the variety of design interest and requirement before recording the macros. The general aspect that needed to be considered for the designer is the information on the requirement of the parametric design, understanding the flow of the macros programming and how to overcome the problem with error from the program line or command.(Ramly et al. 2013)

The advantage of using the CATIA automation is it helps in terms of reducing the cost by improving the efficiencies in drawing and better quality through standardization. While in terms of customization it provides additional functionality which are not provide in the native products. It also help to manipulate the existing function to better fit the customer processes.(Wayzode, 2012)

2.3.1 CATIA INFRASTRUCTURE

For the structure of the program in CATIA, there are three major objects which can be defined as Documents, PartDocument and ProductDocument. All these three objects in CATIA are classes. Even though, the PartDocument and ProductDocument are slightly different as they are classes that gaining properties and process from Document class. Hence, both PartDocument and ProductDocument classes have a similar methods and properties that are obtained from document classes but still having their own unique methods and properties.(Siddesh & Suresh, 2015)

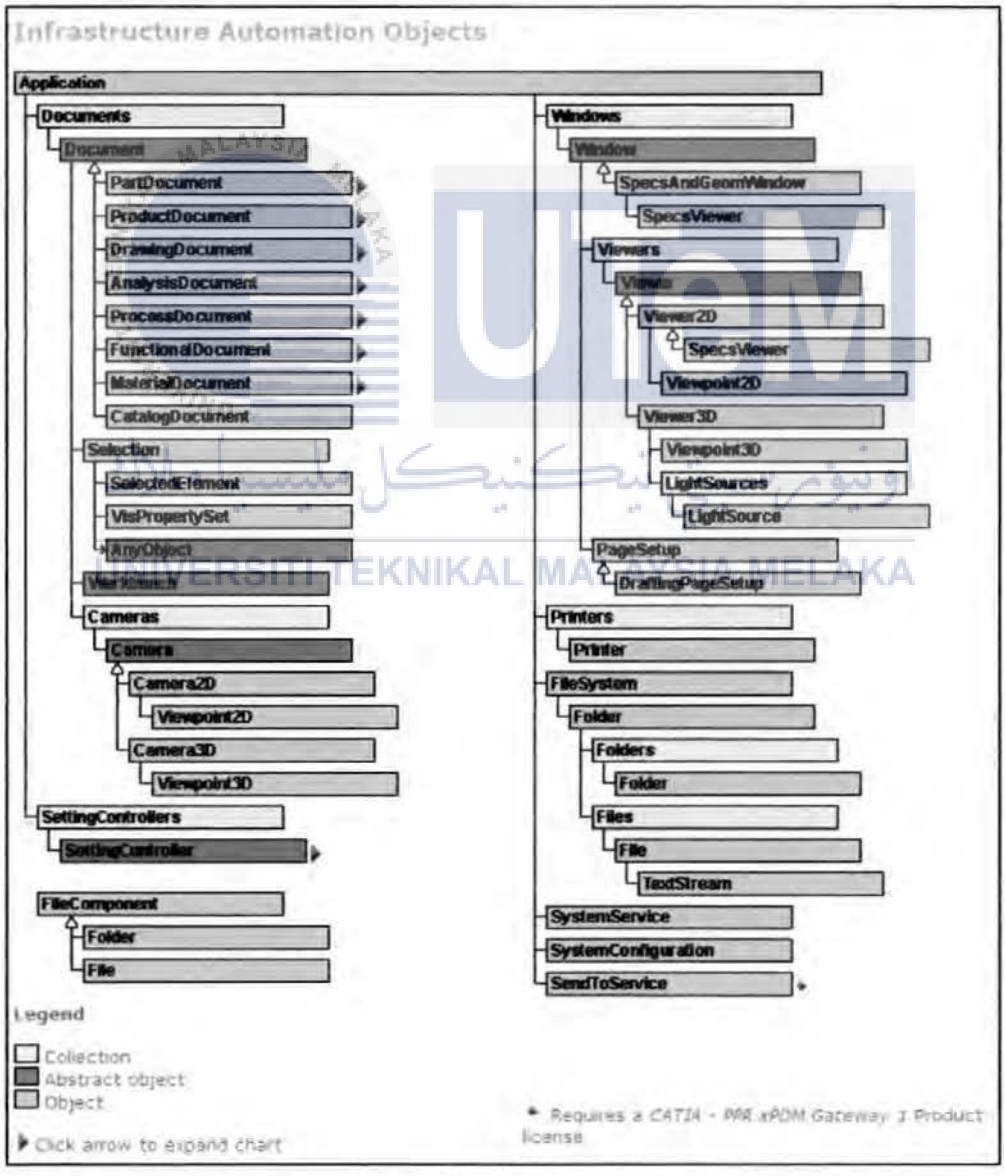


Figure 2-1 Structure of automation object(Duke, 2014)

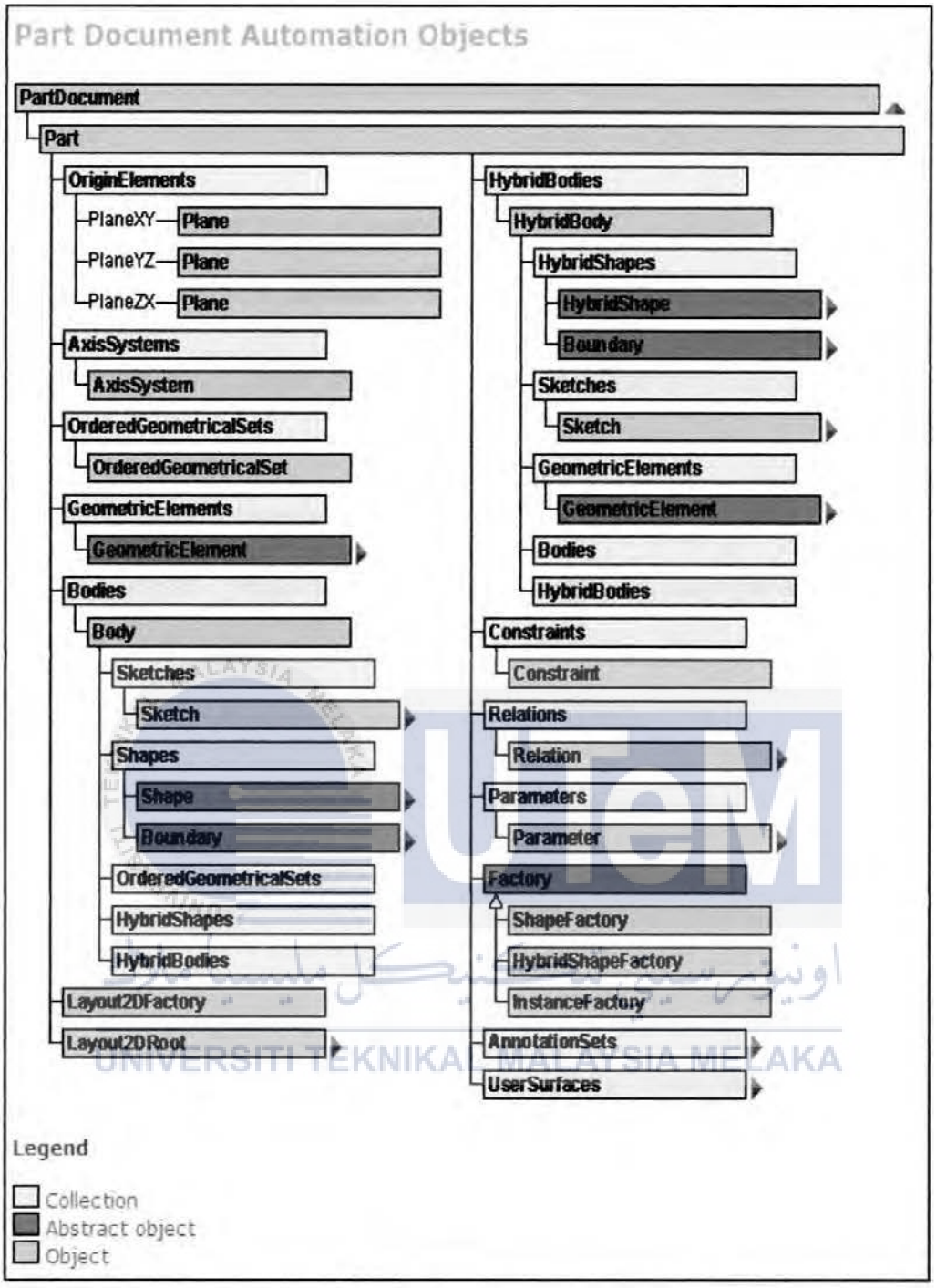


Figure 2-2 Structure of part document(Duke, 2014)

Brief explanation on the infrastucture of CATIA

There are two ways to differentiate the technique to acting on objects which stated as follows:

- Property : It's a characteristic of an object
- Method : It's an action on an object

As visualize in the figure 2.1, there are three types of infrastructure automation objects in CATIA(Duke, 2014) :

- A collection : which is basically is a list of objects.
- An abstract object : which is the object that can't be created concretely but it is needed to be defined the object first.
- An object : an entity that can be created and work with it

The Part Document object as shown in figure 2.2 aggregates or includes the part tree structure starting with the Part object located at the top of the part specification tree. Among these objects, the first objects and collections that displayed and also find in the part specification tree. These objects are:

- The three planes XY, YZ, and ZX you can retrieve from the Origin Elements collection using the Part's Origin Elements property
- The Axis Systems collection
- The Ordered Geometrical Sets collection
- the 3D elements that can be created as reference to position the 3D objects and stored in a Geometric Elements collection which can be retrieved using the Geometric Elements property(Siddesh & Suresh, 2015)

- The created bodies, which starting with the main body and stored in the Bodies collection that can be retrieved using the Part's Bodies property. These bodies contain the part's geometry.
- The hybrid bodies that created, which stored in the Hybrid Bodies collection.

In addition, the Part object aggregates:

- The constraints which it can be set to the 3D objects and stored in the Constraints collection. It then can be retrieved using the Part's Constraints property
- The relations between parameters stored in a Relations collection can be retrieved using the Part's Relations property
- The parameters stored in a Parameters collection can be retrieved using the Part's Parameters property(Wayzode, 2012)
- The factories: a Shape Factory object to create shapes, a Hybrid Shape Factory object to create hybrid shapes, and an Instance Factory object to instantiate either User Defined Features or a Power Copies. The first two factories can be retrieved using the Part's Shape Factory and Hybrid Shape Factory properties respectively.
- The functional annotations and tolerance objects aggregated under the Annotation Sets and User Surfaces collections.(Bilalis, 2000)

The Bodies collection includes Body objects, which is the one being the main body that return by the Main Body property of the collection. The part has in addition a current body and a current shape, returned or set using the Current Body and the Current Shape properties of the Part object respectively.

The current body is the body in which a new shape is added when using the Shape Factory object. The current shape is the shape of the current body after which a new shape is added in the specification tree when using the shape factory. The user then need to make current the appropriate body and shape before using the shape factory.(Bernard, 2003)

The other collections can be classified in two categories:

1. The collections that only contains objects and have methods to retrieve and possibly remove them, but leave the dedicated factories for object creation, such as Sketches, Geometric Elements, and Shapes
2. The collections that also have methods to create the objects they contain, such as Constraints, Relations, and Parameters.(Duke, 2014)

2.4 MACRO PROGRAMMING IN AUTOMATION

One of the application for automation can be done by approaching the fundamental of Macros tools in CATIA V5 software. CATIA V5 is one of the open system for developing advanced Macros for a special situation. A macro is a script of programing code written in a certain programming language which groups a set of operations or command that define a certain task. These Macros may be useful for creating, analyzing, measuring, modifying, translating, optimizing surfaces, solids, wireframes and many more possible command. Macros are also useful for assembly operations, CAM operations and all multidisciplinary applications. Macros are developed using Visual Basic Script (VBScripts) and Visual Basic Advance (VBA). In order to develop a macro in CATIA all its need is the inputs, outputs and necessary supporting data from the user.(Wayzode, 2012)

A macro uses a text string that combines commands and options for repeated use. It is a method to obtain command-line input and storing it for later use through the usage of macro recorder. Thus, the generation of a model task that would take a long series of actions can be accomplished just by a one click operation. The operation of macro can be as simple as calling a command and providing a single input, or a complex series of commands and options.(Autodesk, 2008)

Macro for CATIA V5 can be piloted with mainly three possible languages as follows:

- CATVBS
- CATVBA
- CATScript

2.4.1 VISUAL BASIC (VB)

CATIA V5 automation was originally designed for VB6, VBA, and VBScript. But over the years, VB6 has been replaced by VB.net and Microsoft no longer officially supports it which is supported by CATIA V5 R16 and onwards. VB6 is more complex but more powerful than VBA, as is VBA over VBScript and CATScript. CATIA Macro languages supported by CATIA which been focused are VBScript, CATScript, and VBA, where all those language is a derivatives of Visual Basic used in scripting. (Halvorson, 2010)

Visual Basic is a tool that allows the user to develop a Graphic User Interface (GUI) application for windows system where the applications have a familiar appearance to the user. Operation of Visual Basic is by event driven process which means the code remains stationary until the command is executed. Example of this case is defined by the operation of button pressing and menu selection. Visual Basic also instructed by an event processor. Nothing will happen until an event is detected. Once an event is detected, the code corresponding to that event is executed. Program control is then returned to the event processor.(Tylee, 1998)

2.4.2 DEFINITION OF CATSCRIPT, CATVBA AND CATVBS

CATScript, CATVBA and CATVBS are type of languages in programming. Those languages work with object and methods. An object is defined as container that stores collected information where the information could be a CATPart, a line, or a surface. While method terms means a command from which an object is created or modified or also from which information is read. (Ziethen, 2013)

i) VISUAL BASIC FOR APPLICATION (VBA)

VBA (Visual Basic for Applications) is a subset of Visual Basic and is hosted in applications such as CATIA prior to V5R8 and Microsoft Word. With the VBA, it helps in giving a complete programming environment with an editor, debugger, and help object viewer. It also allows the function to declare the object library used. (Tylee, 1998)

In CATIA, VBA has the full VB6 syntax and Integrated Device Electronic (IDE), which is similar to VBA in Excel. It operated by event driven process which means GUI oriented, and has full IDE, but it still cannot execute a program without the host application running. The benefits of using CATVBA Macros include using the GUI, building forms, and the ability to debug the macro editor. While the downside includes a problem where the programs cannot be compiled into executables or DLLs (Dynamic Linked Libraries) and they do not run in their own memory space. (Weisenberger, n.d.)

ii) VBSCRIPT

VBScript is another variation of the Visual Basic Programming language and is very similar with the VBA programing language. All elements of VBScript are present in VBA, but some VBA elements are not implemented in VBScript.

VBScript was originally designed to run in Web applications such as Internet Explorer. One of the benefits of VBScript is that it is written in plain ordinary text which means the code can be written in a simple software such as notepad. CATIA objects can be called but no type is used as the system tries to dynamically call methods and properties of objects.

It can be used on both Windows and UNIX versions of CATIA. The disadvantage of VBScript is it is slow, is limited for interface development, and has the least functionality. The file extension is ".catvbs".(Tylee, 1998)

iii) CATSCRIPT

CATScript is portable version of VBScript owned by Dassault Systèmes' and it is a variant of the VBScript. Prior to CATIA V5R8 edition, CATScript Macros is design to run on UNIX and Windows systems. It is a sequential programming language and non-GUI oriented. Coding for CATScript can also be created by using a regular text editor like notepad.

Advantages of writing CATScript Macros include free to use, macro recording, personal time saving operations, and rapid deployment. The disadvantages of CATScript are limited flexibility and difficult to debug.The file extension is ".CATScript".(Tylee, 1998)

2.4.3 MACRO METHOD FOR GENERATION OF CAD MODEL

Both CATScript and CATVBS explicated the language that serve as the foundation for programming in Macros for CATIA V5. Creating a programming environment in CATIA with the CATVBA offers more potential for CATIA V5. CATVBA has a compiler and offers many tools for designing and these differentiated it from CATScript and CATVBS. An overview of these three language is shown in table 2.1.

Table 2-1 Overview of macro language in CATIA(Tylee, 1998)

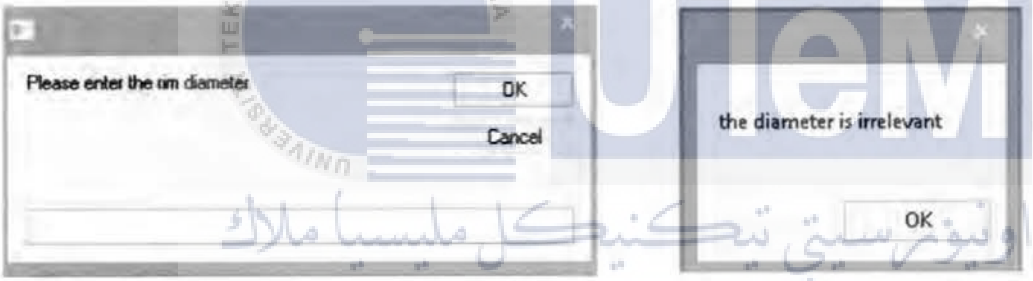
LANGUAGE	FILE FORMAT	DESCRIPTION	APPLICATION
CATScript	*.CATScript	Reduced VBScript interpreter (Windows,UNIX)	Macros (windows and UNIX), CATIA Knowledgewere
CATVBS	*.catvbs	Complete VBScript interpreter (Windows,UNIX, since V5R8)	Macros (windows and UNIX), CATIA knowledgewere
CATVBA	*.catvba	Visual Basic for Applications Compiler (Windows)	Menu based application (Windows)

Automatic generation of cad model in Macros are focusing on two main applications which is the *Userform* and the automation process

i) DIALOGUE BOX

Macro can be interactive with the user through the application of the dialogue box in VB programs. The dialogue box is the custom user interface on screens that can be developed in VB. Creating a dialogue box is an important feature in Visual Basic program as it offers the ability to easily and quickly create graphical user interface (GUI) for the VB programs.(Ross, n.d.)

The fundamental of any GUI in a VB program is a "Form". Form control are objects which can be place on Excel worksheet or *Userforms*, which give the functionality to interact with the data. With the CATScript it allows the user to program a dialog box with text communication between a user and a macro. The function is named MsgBox for output data and InputBox for input data.



(a) (b)

Figure 2-3 Example of (a) inputbox and (b) messagebox

These forms help the user to interact with the model. The form contains text boxes, labels and a command button. Labels are used to show the message or to identify the parameters needed to edit in the text box. Text box is used to take the parameters to execute in the program. Command button is used to run the intended task.

For more complicated and multiple task to be applied to the object, there is a function tool called *Userform*. In the VBA editor the user can choose "Insert" option and then

then "*Userform*". Then an empty form will be shown where the user can add controls like textboxes, checkboxes, lists, comboboxes, command buttons and much more. The *Userform* and all the controls that has been put into it all have a series of standard events, which a code can be inserted into it. That could be what to do, when the user clicks on the control, presses a key and much more. The visualization of *Userform* in visual basic editor are shown in figure 2.3 below.

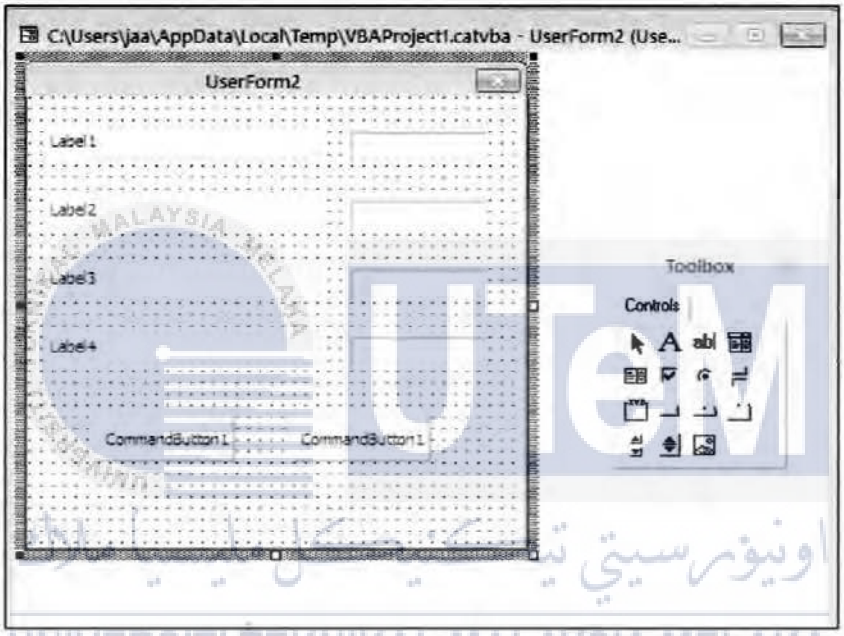


Figure 2-4*Userform* interface

2.4.2.2 AUTOMATION PROCESS

To properly define the Automation process, an example of a generation process of a car rims model can described the automation process. The automation can be done by using the macro recorder during the creation of the model. The modeling process of the car rims need to be create step by step without using the undo button during macro recording. When the modeling process is finish, stop the recorder and edit the program in the macro editor by applying the required parameter into the program as shown in the figure 2.4 below. The programming script obtain from the result of macro recording is shown in figure 2.5.(Siddesh & Suresh, 2015)

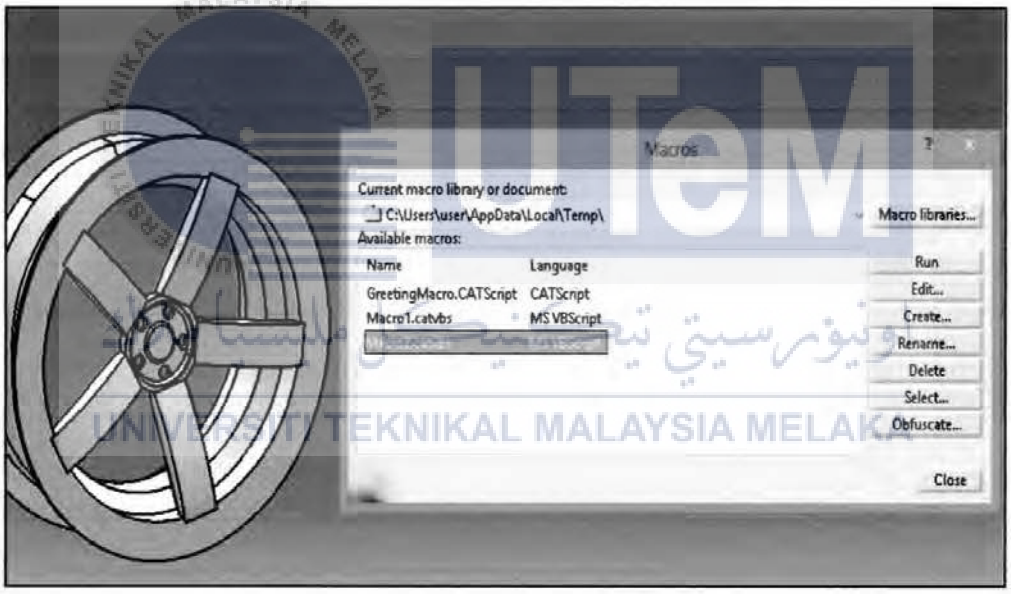


Figure 2-5 Example of the macro recording process

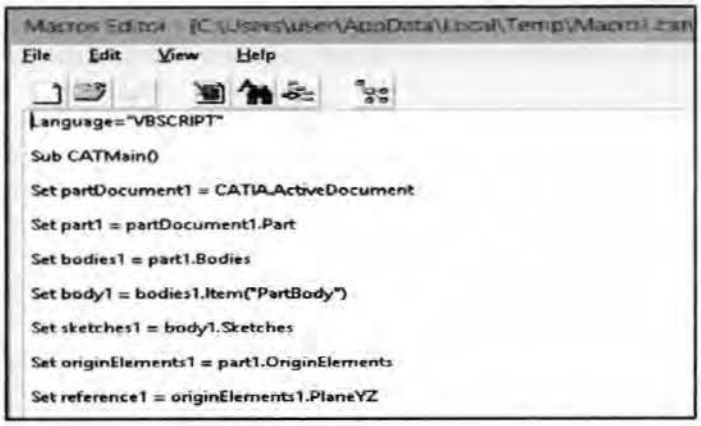


Figure 2-6 Sample of macro recorded programming

2.5 RELATED RESEARCH

In accordance with this article, a lot of effort has been put in previous research in order to automated modeling in CATIA and different 3D CAD software"s. Wrobel J. et al. was researching a method to describe customization tools that already that already available in CATIA V5 system. In addition of that, during the development process, they have developed a system for parametric model of mould tool in CATIA V5 by the application of VBA scripting. Other researcher, Shah D. B. has invent a method to develop a parametric model for automated modeling of flange coupling. The method that were used by him was by using the Autodesk Inventor integrate with Microsoft Excel in purpose to produce 3D modeling and 2D drafting modeling of flange coupling automatically.(Shah D. B. 2010) Other researcher group Cukovic S. et al has created a macro programming in purpose to determine the differentiation between profiles of helical surfaces and explained the principles for complex surface modelling process. The macro method that were used will optimizes the time turning from hours to seconds, increase flexibility, and possibility to use of unlimited number of times.(Cukovic et al. 2010) From these literature review, it proves that CATIA is a suitable platform for creation of macro to make automation modelling process to be done.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

In previous topic, the definition of CAD automation has been explained. The CAD automation drawing focused on the ability to generate a model automatically. Therefore, further discussion on how to develop the automation of CAD modeling will be discussed in this chapter. Apart from this processes, an attempt on VBA programing will be presented. On the other hand, Macros programming is a method where the configurations of the design are done by recording a script of commands, editing the data value and adding a certain new command line to create a new model. The steps and the procedure to create a macro programming are discussed in this project.

3.2 AUTOMATION METHODOLOGY

Every resources for information and research data collected in order to progress further to complete this project have been gathered and referred from journals, thesis, internet and lecturer. The main process of automation is divided into two main phase which is:

- The modelling process
- Macros programing

The flow of the main process is shown in figure 3.1. These items are main key in automation in CAD. The development process of CAD automation are shown in figure 3.2 to deliver a better understanding of the flow process of the project.

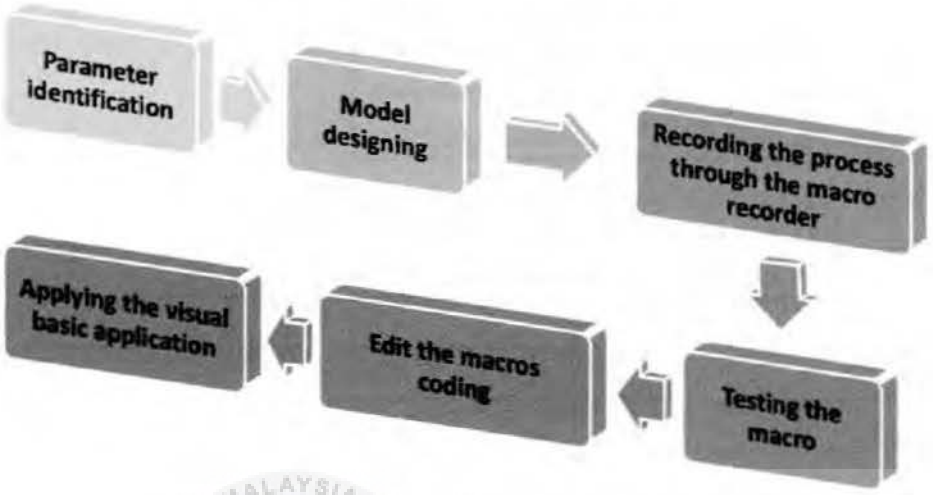


Figure 3-1 Flow chart of main process of automation

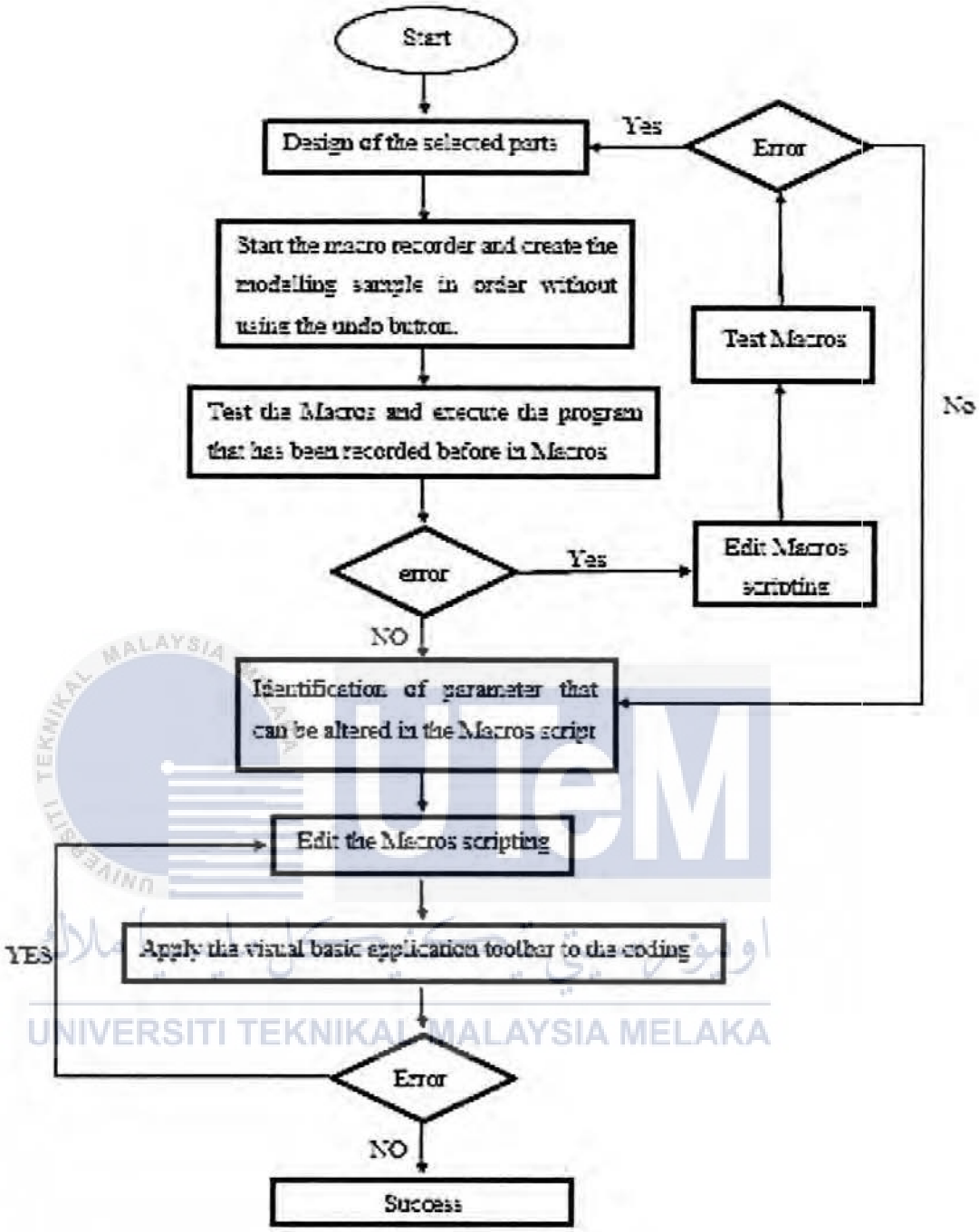


Figure 3-2 Flow chart of automation development process

3.3 CAD AUTOMATION CONSTRUCTION

In this project, there are four main tools and apparatus that will be used in order to complete the CAD automation process. These items are shown as follows:

1. Three models of car rim
2. Personal computer
3. CATIA V5R21
4. Macros tool

3.3.1 CAR RIMS MODELS

For research purpose, three different design of car rims has been decided to be the model design for the automation process in this project. Basically the design of the rims are based on the real rim design manufactured by the Vossen company. To be more specific, the Voseen rim model that were used is CV3-R model listed under Monoblock CV Series.

Figure 3-3 CV3-R Vossen rims

The CV3-R utilized a low-pressure casting process to create a single piece mono-block wheel with the light weight properties. It is manufactured in two gloss finishes for the appearance. The general size available for CV3-R rim are 19", 20" and 22" diameters and a variety of bolt patterns and offsets. CV3-R possess the symbolic concave design for the rim spoke. The CV3-R usually offer five piece of the shaft spoke.

Based on this design, the idea to make the design can be altered has been implied to the model. To make the design has an ability of alteration, several important aspects are needed to be considered in the process of creating the model. The first one is to identify the original parameter of the actual design. This is important because before the automation process begin, a sample design is needed to make as a benchmark level for the alteration parameter in model so that it can be edited in the future.

3.3.2 COMPUTER UNIT AS PROGRAMMING PLATFORM

The requirement to finish this project is including a fully functional computer system. A computer unit plays a huge role in order to execute CATIA software and Macro programming. The computer must have a compatible specification that fulfill the requirement to run the CATIA software.

Table 3-1 minimun requirement recommend to run CATIA V5(Duke, 2014)

Supported platforms	Windows XP professional or higher
Minimum requirements for the graphic	• OpenGL 2.0 or higher • A screen resolution of 1280x1024 pixels is required at least • Texture hardware is highly recommended (NVIDIA / AMD ATI). • The colour depth must be set to 24 bit at least
Memory requirement for RAM and swap space	• Requires at least 8 GB (64 bit) Main Memory. • The total amount of Main Memory + Swap Space must be approximately 8 times larger than the size of the data base/scan file to be used.

3.3.3 CATIA V5 AS PLATFORM FOR DESIGN MODELING

The generation of the model will be done by using CATIA V5 software. CATIA V5 is a root for CAD automation in this project as it will be the example of CAD automation platform. The selected model of CV3-R rim car which have been discussed in previous topic will be design in the CATIA V5. The process initiated by designing the model first in CATIA software based on the acquired parameter from the actual model of CV3-R rims car. The design will be created manually until the desired model are acquired where it will be a sample model for the reference that will be used in later macro creation. To start the modeling of the sample model, from the menu toolbar select Start, then select the Mechanical Design and then choose Part Design. A menu box will appear and clicking on ok button will begin the modeling process.

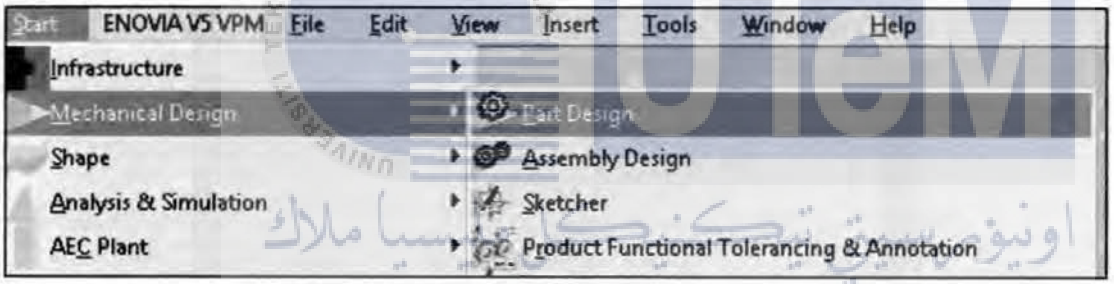


Figure 3-4 Part design from the menu toolbar of CATIA

The first step in the modeling process of the car rims start with the construction of two circle with same origin to be the base of the rim outer diameter. The constraint for the outer circle dimension must be put in as this parameter will be used in the future automation process. While for the inner circle, the dimeter constraint are not needed to be included. But the distance to the outer circle must be put in so that the position of the inner circle will fix to the outer circle as shown in figure 3.5

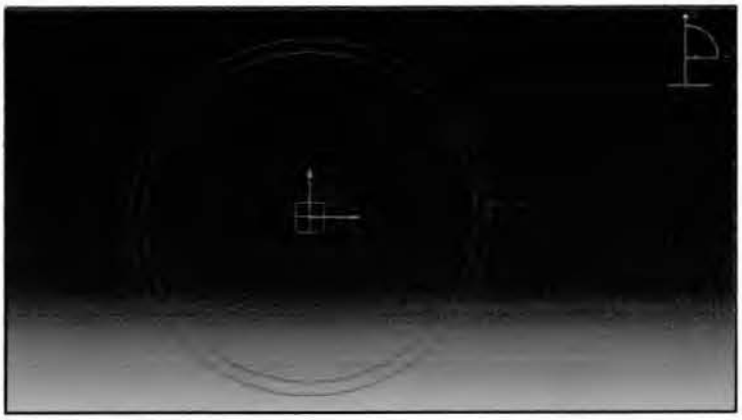


Figure 3-5 circle drawing process in CATIA

Then the circle is extruded through the application of pad and shaft so that the basic ring for the rim is created as shown in figure 3.6. The pad process must be done with mirror option so that the center of the plane are place symmetry at the middle of the rim.

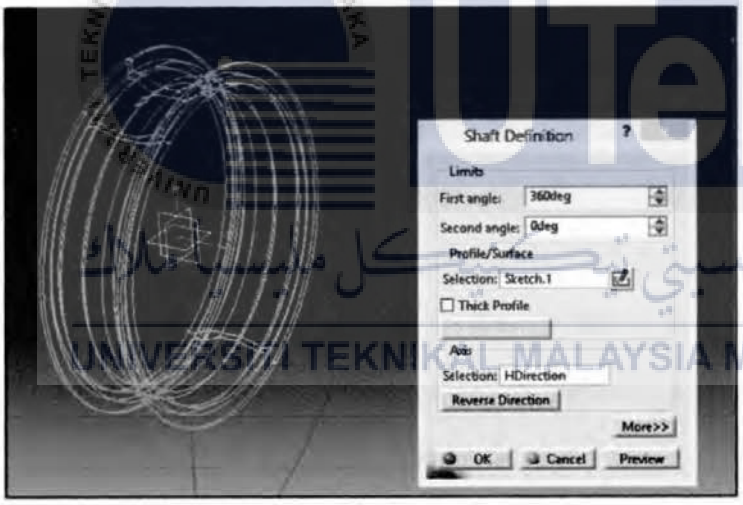


Figure 3-6 pad process and shaft process

Then a plane is offset from the origin at one side of the rim to make a base plane for the center rim holder. A smaller circle is draw on the plane and extruded to make the center rim holder. The surface of the middle circle will be the reference plane for the spoke of the rim as shown in figure 3.7. The end of the spoke must be coincidence to the inner circle of diameter rim so that the length of the spoke is fix to diameter of the rim.

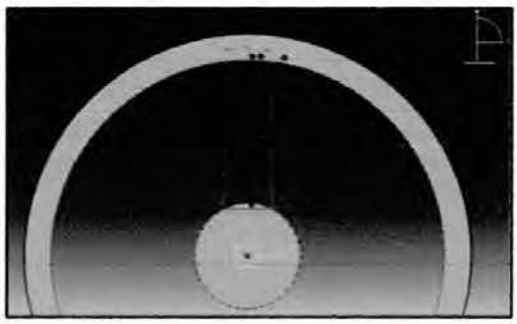


Figure 3-7 drawing of rim spoke

Extrude the spoke of the rim and then use the circular method command to symmetrically copy the number of the spoke around the rim as shown in figure 3.8

Figure 3-8 copying the spoke symmetry with circular method

At the final stage, the circle for the bolt hole is draw on the center rim holder and the pocket command is used to create the hole. Then the circular method is applied to the pocket so the the position of the bolthole is symmetry around the rim.

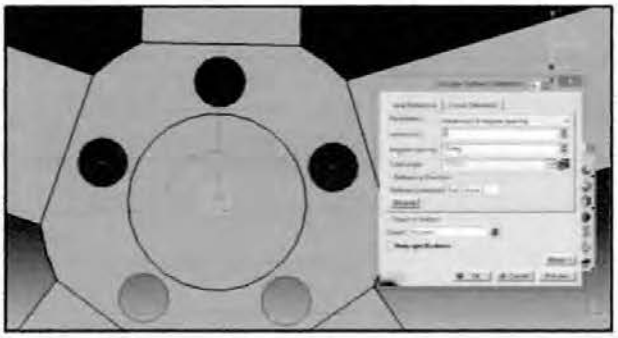


Figure 3-9 circular method for bolt hole

Process that will involve during the creation of the model includes the drawing of the circles and line in the design sketches, circular method, pad and pocket definition, groove and shaft creation and parameter constraint. Figure 3.10 shows the finish product for the sample model.

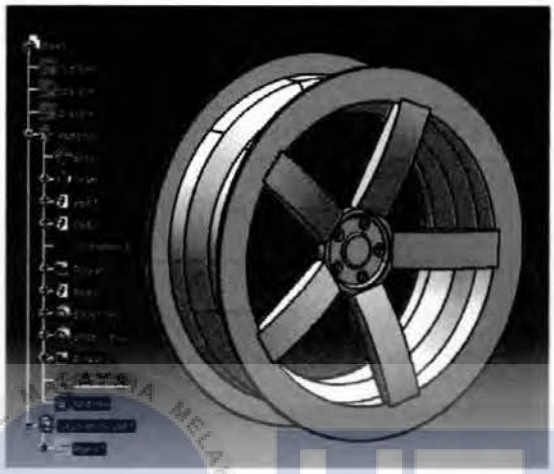


Figure 3-10 Finish product for CV3-R car rim model

By refering to the sample model that has been created, open a new part design drawing to create a new design. But before initiate the part drawing, choose the macro command under tools command bar and then choose start recording option as shown in figure 3.11.

Figure 3-11 Execution of macro from menu toolbar

When executing the macro recording tools, by referring to the sample model the creation of the new model must be created only in one direction. It means that while modeling the new model, a mistake must be avoided because the process cannot be undone as it will affect the result of programing script that are being recorded.

An icon for stop recording will appear when the macro recording is executed. To stop the recording click the stop recording icon or choose the stop recording option under macro command bar as shown in figure 3.12.

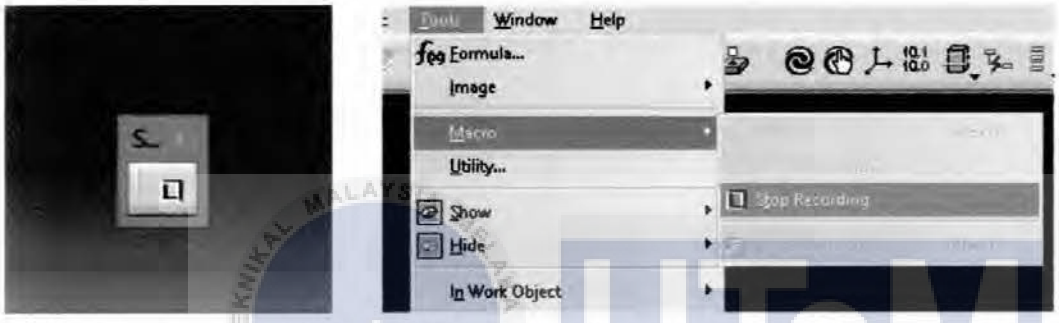


Figure 3-12 Command to stop macro recording

The application of macro recording process help the user to obtain the programming script automatically for the macro. The recorded macro needed to be run first to check if there has any error. If there are no error occurred, the recorded macro later can be used for editing process. If else there is an error, the design need to be redesign until there are no more error.

3.3.4 MACROS PROGRAMMING IN CATIA

As stated in previous chapter, a macro is a text string that combines commands and options for repeated use. In order to develop a macro in CATIA V5 all its need is the inputs, outputs and necessary supporting data from the user. Macro programming help the user to create a programing script in VBA that will be used for the automation process. One of the application to directly obtain the programming script in macro is by using the macro recorder. As mentioned in previous sub topic, to record the macro user's need to create the model following the proper step continously with using the undo button. The recorded programing script can be use to generate another model automatically by executed the programing in Macros shown in figure 3.13.

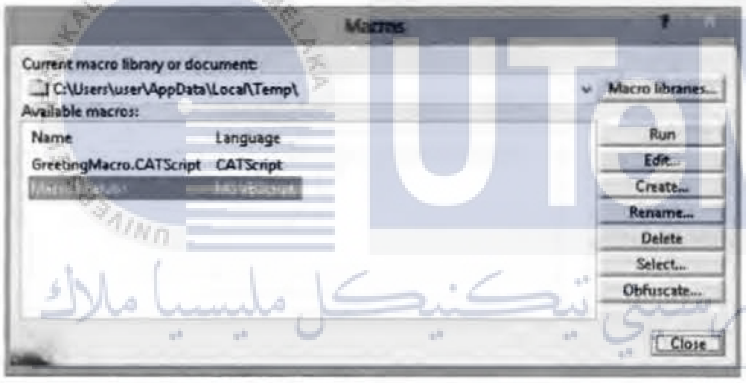


Figure 3-13 Macro menu

This programming script also can be edited to make the model more interactive with the user by adding the user interface unit into the model. Thus user can easily adjust the model by reffering to the sample model that has been recorded. Following the example of sample of the CV3-R model from before, a user interface can be added so that some of the parameter can be changed directly.

3.3.4.1 DESIGN PARAMETRIC IDENTIFICATION

To implement this concept, the user need to identify the suitable parameter that can be altered for the new design. As an example in this case study, the model that been used as reference is the car rims. Thus the common parameter that can be altered for car rims will be identified so it can be used for editing the programing script. The figure below shows the parameter used for common customization of the car rims. The list in figure 3.14 and table 3.2 shows the suitable parameter required for a CV3-R rim design alteration.

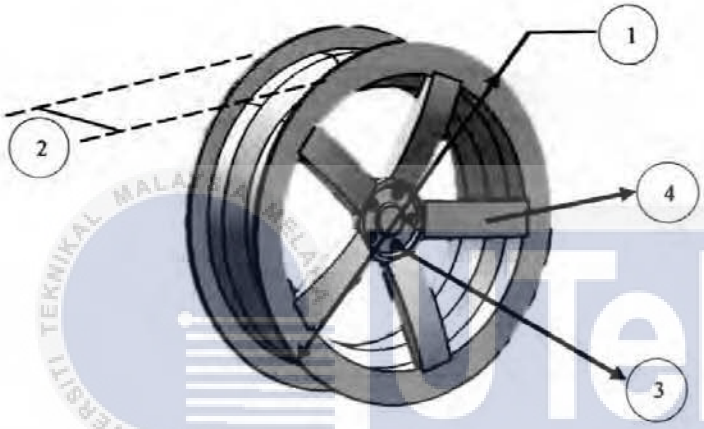


Figure 3-14 Ilustration of common parameter for rim custamization

Table 3-2 Suitable parameter for editing process

No.	Parameter Characteristic
1	Rim diameter
2	Rim width
3	Number of bolt hole
4	Number of rim spoke

3.3.4.2 MACROS PROGRAMING PROCEDURE

After the requirements of the design have been set, the editing process of the macro can be done. The Macros are edit by adding a dialog box functions into the programing script. These dialogue box are categories into two function which is inputbox for input data and messagebox for output data. These dialogue box are used to create a pop-up message box which allow the user to enter the necessary input data for the model. The figure 3.15 below shows the application of the dialogue box into the programing script for the car rims model.

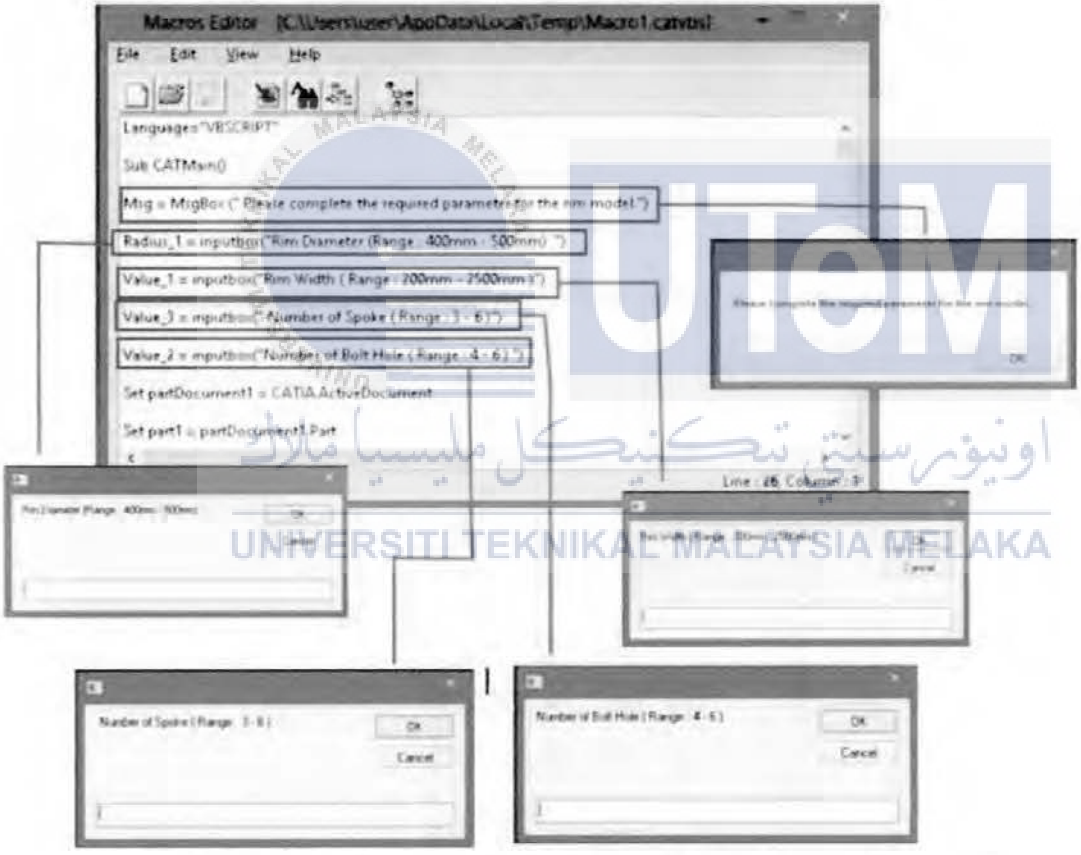


Figure 3-15 Application of inputbox and messagebox in macro

To make the command task can be compiled into one form for the input value, the usage of *Userform* will be applied to the Macros coding. To show the *Userform*, it can be access through the visual basic editor or the user can call the function by writing the following command in the CATIA command box: *MyForm.Show*

That is as an example if the user has named the *Userform* "MyForm". To add functionality into the form, use the buttons in the Toolbox to draw controls on the form. The Toolbox identifies the different controls function that can be added to the form. The toolbox can be customized in many ways including the following:

- Add pages to the Toolbox.
- Move controls from one page to another.
- Rename pages.
- Add other controls, including ActiveX controls, to the Toolbox.
- Copy customized controls from the form into the Toolbox.

Figure 3-12 The workbench of the toolbox

To make the button assigned to the form function properly, a specific code in VBA language must be inserted into the functionality of the button. For example, a code been added a to the button's click procedure of th Command Button to make it deactivated the *Userform*. The code is write and shown as follows:

Private Sub CommandButton1_Click()

Unload Me

End Sub

However, before the command "Unload Me," the user can write a coding script that does something with the user input. To make something happen when the *Userform* opens, in the form's Initialize procedure the code should be visualize as follows:

Private Sub Userform_Initialize()

'Action code' ⟶ Coding script for action to do when the form opens. For instance, this could be input values from textbox into the model parameter

End Sub

The function of the created Macros were tested in order to check wether the Macros will works or not. To ease the opertion by the user, a toolbar icon is created using the Macros programming. A button can be assigned to the toolbar as a shortcut to executed the macro.

CHAPTER 4

RESULT AND ANALYSIS

4.1 INTRODUCTION

In this chapter, focus will be given on the development of the Visual Basic coding by using the Macros software in CATIA. All the coding will be defined for better understanding. Other than that, results obtained from the Macros to control the requirement for the automation of the drawing in CATIA also will be discussed. In addition, all the parameters that connected the *Userform* function to the model drawing will be discussed.

4.2 DEVELOPMENT OF CODING FOR 3D MODEL OF THE CAR RIMS

In order to make the modelling process of a model sample able to be automated in CATIA, a set of coding logarithm for the model need to be defined in the first place. To get the coding script, the modelling process need to be recorded by using the function of start recording in the Macros tools. The macro will be the module for the coding reference object.

Figure 4-1 Recording of the macro menu

The modelling process must be fluent without using the undo or delete button while recording in macro to avoid the complexion of algorithm in the coding. Once the drawing finish, stop the recording before exiting the drawing thus all the data from the start of the drawing until finish will be stored in the Macros. To show prove the theory, three design of different car rims has been choosing to be created as the model as shown in the figure 4.2.

(a) (b) (c)

Figure 4-2 (a) Design 1 (b) Design 2 (c) Design 3

The coding script of these models are stored in the Macros library and can be run again to automatically create the same model by selecting the code in the library and click the 'run' button. These programing script will be edited to be use for the later in depth process of automation. The example of part of the coding script for design 1 are describe in the details shown as follows:

```
Sub CATMain()
Dim documents1 As Documents
Set documents1 = CATIA.Documents
Dim partDocument1 As PartDocument
Set partDocument1 = documents1.Add("Part")
Dim part1 As Part
Set part1 = partDocument1.Part

Dim bodies1 As Bodies
Set bodies1 = part1.Bodies

Dim body1 As Body
Set body1 = bodies1.Item("PartBody")

Dim sketches1 As Sketches
Set sketches1 = body1.Sketches

Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements

Dim reference1 As Reference
Set reference1 = originElements1.PlaneYZ
```

Recorded macro code to declare the references for the drawing of the model.

```
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = 0#
arrayOfVariantOfDouble1(3) = 0#
arrayOfVariantOfDouble1(4) = 1#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 0#
arrayOfVariantOfDouble1(8) = 1#
```

Process to declare the variable of "double" (floating point double precision). The index is dimension with counting started at "0".

```
Set sketch1Variant = sketch1

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
```

Process to defining the axis of the drawing

```
line2D1.ReportName = 2
Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")
line2D2.ReportName = 3
Dim circle2D1 As Circle2D
Set circle2D1 = factory2D1.CreateClosedCircle(0#, 0#, 400#)
```

Example of body design process code. This one is the creation process of a circle for the

```
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
```

Updating the process every time there is a changes in the process

```
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory
Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, 20#)
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
Dim length3 As Length
Set length3 = limit1.Dimension
length3.Value = 200#
pad1.IsSymmetric = True
part1.Update
```

Pad process coding script

```
End Sub
```

This coding script function is to declare the acquired code through the recording Macros function. The recorded code is important in order to create a relation with the coding from the *Userform* later on so that the automation process of the model can be done successfully. Also, the parameter from this code will be the object references for the *Userform* coding.

4.2.1 PARAMETER ANALYSIS

Before the coding process for automation can be done, each parameter that correlated with the function that can be altered for the modification of the model need to been identified and listed first. As discussed in the previous chapter, there are four common parameters that can be altered for a car rims as listed as follows:

- Rim diameter
- Rim width
- Number of bolt hole
- Number of rim spoke

This parameter can be acquired from the original coding script gain from the recorded macro of the rims model. As for an example taken from the part of coding of design one, the coding from line Macros script shows the dimension of rim diameter. The coding example from the script are shown as follows:

```
Dim length1 As Length
Set length1 = constraint1.Dimension
length1.Value = 431.8
```

Declaration of length1(diameter of rims)

Coding for the initial value of the diameter.

This is the line of coding that shows the size of the rims diameter. The parameter from this line which is *length1* is need to be identified and declared so that it can be synchronized with the application of *Userform*. Other parameter that can be identified are shown as follows:

Coding parameter for rim width

```
Dim length3 As Length
Set length3 = limit1.Dimension
length3.Value = 177.8
```

Coding for the initial value of the rim width

Coding parameter for number of spoke

```
Dim angularRepartition2 As AngularRepartition
Set angularRepartition2 = circPattern1.AngularRepartition
Dim angle1 As Angle
Set angle1 = angularRepartition2.AngularSpacing
angle1.Value = 72#
intParam1.Value = 5
circPattern1.ItemToCopy = pad4
```

Coding for the initial number of spoke

Coding parameter for number of bolt hole

```
Dim angularRepartition4 As AngularRepartition
Set angularRepartition4 = circPattern2.AngularRepartition
Dim angle2 As Angle
Set angle2 = angularRepartition4.AngularSpacing
angle2.Value = 90#
intParam2.Value = 4
circPattern2.ItemToCopy = pocket1
```

Coding for the initial number of bolt hole

4.3 DEVELOPMENT FOR THE APPLICATION OF *USERFORM*

The application of the *Userform* in Visual Basic is one of the main roles to execute the process of CAD automation for CATIA V5. The *Userform* can be created from the Visual Basic Editor which can be access from the toolbar in CATIA. The application such as command button and the text box can be assign manually into the user form through the function of Toolbox. For each application assigned to the *Userform*, the code need to be put in manually to make it function. The coding developed in the *Userform* is the key item used to link the logarithm between the coding of the *Userform* and the coding of the model. The interface shows an empty form to be fulfilled by the user to complete the automation process. For this project, the *Userform* used are divided into two phase which is the first phase is to choose the design and the second one is for the parameter input of the design.

The *Userform* entitled *Rim Design* will shows the picture of different selection of the available rim design with a command button assigned to each picture. At the bottom of the *Userform* there will be a command button with a 'cancel' function. Figure 4.3 shows the image of the *Rim Design Userform*.

Figure 4-3 *Userform* for design variety (first phase)

The sample coding logarithm for application of the command button to choose the design are shown as follows:

```
Private Sub CommandButton1_Click()
Unload Userform1
Userform2.Show
End Sub
Private Sub CommandButton2_Click()
Unload Userform1
Userform3.Show
End Sub
Private Sub CommandButton3_Click()
Unload Userform1
Userform4.Show
End Sub
Private Sub CommandButton4_Click()
Unload Userform1
End Sub
```

Example of command code use to declare the button clicking action on command

Unload command uses is to close the userform1.

Userform.Show command uses is to call the parameter input form respectively according to the design. (userform1 = design 1, userform2 = design 2, userform3 = design3)

Figure 4-4 Form for the parameter input

For the second phase, the form will include empty textbox with the description for the parameter requirement as shown in figure 4.4. It also includes two command button with the function to cancel and create. To make the data input can be use into the model, the coding line for parameter of the model need to be changed. The value of the parameter in the coding are replaced with the *textbox* command function. Thus the value for the parameter can be read and gain from the value put in the textbox of the Userform. For example, taken the coding sample from the design one as shown from the previous topic, the original coding value for the rims diameter is *length1.Value = 431.8.* From this line, the number can be change to the *textbox1* command which is the empty textbox for the rim diameter value. This will allow the value of the rim diameter to be altered depends on the input value from the textbox. The sample coding are shown as follows.

Parameter for rim diameter coding

Dim length1 As Length

Set length1 = limit1.Dimension

Length1.Value = textbox1

The value of the diameter is changed to the textbox 1 command input from the userform

Parameter for rim width coding

Dim length3 As Length

Set length3 = limit1.Dimension

length3.Value = textbox2

The value of the width is changed to the textbox 2 command input from the userform

Parameter for number of spoke coding

Dim angularRepartition2 As AngularRepartition

Set angularRepartition2 = circPattern1.AngularRepartition

Dim angle1 As Angle

Set angle1 = angularRepartition2.AngularSpacing

angle1.Value = 360 / textbox3

intParam1.Value = textbox3

circPattern1.ItemToCopy = pad4

Value for number of spoke is changed to textbox 3 and total angle is divided with number of spoke to make it simmetry

Parameter for number of bolt hole coding

Dim angularRepartition4 As AngularRepartition

Set angularRepartition4 = circPattern2.AngularRepartition

Dim angle2 As Angle

Set angle2 = angularRepartition4.AngularSpacing

angle2.Value = 360 / textbox4

intParam2.Value = textbox4

circPattern2.ItemToCopy = pocket1

Value for number of bolt hole is changed to textbox 4 and total angle is divided with number of bolt hole to make it simmetry

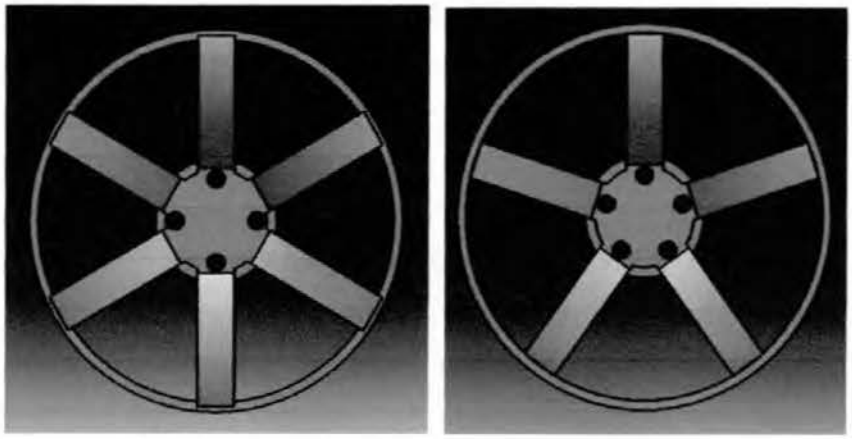
4.4 SIMULATION AND RESULT OF THE PROGRAM

In this part, the final Macros logarithm that has been finish edited is test by assigning the Macros to a button icon as a shortcut to launch the program. When the Macros is executed, a form entitles *RIM DESIGN* will pop out to show the menu for the selection of the three available design as visualize in figure 4. Clicking either *DESIGN 1, DESIGN 2* or *DESIGN 3* button in the form with the mouse will close the form and result in another form to shows up. This new form will show a list of empty text box with the parameter description that need to be filled in as shown in figure. The value for the input must be the number within the range as given in the form or else an error will occur during the automation process. When the desired parameter of the design has been filled, clicking on the create button will create the design corresponding with the value that has been filled in the textbox.

To shows the flexibility of the created Macros program, the different parameter value for design 1 is set to be used into the program as shown in the table 4.1. The result for the object created from this value is shown in figure 4-5. Some of the example of the possible rim design that can be created with this program is shown in the figure 4.6. This results prove the logarithm of the automation is function successfully.

Table 4-1 Different parameter of Design 1

No.	Parameter Characteristic	Design 1 (A)	Design 1 (B)
1	Rim diameter	400mm	430mm
2	Rim width	200mm	200mm
3	Number of bolt hole	4	5
4	Number of rim spoke	6	5



(a) (b)

Figure 4-5 (a).(b) Example of generating a model with different parameter

Figure 4-6 result for the different design and parameter

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 INTRODUCTION

In this chapter, the topic will be focus about the overall conclusion of CAD automation. The advantages and application of CAD automation will be conclude.

5.2 CONCLUSION

As a conclusion to this project, Macros application and visual basic programing played a huge part on developing a CAD automation process in CATIA software. The automation process are meant to help in reducing the development time of a product especially in mass production industry and also minimizing error. Macros tools is one of the system for automation process development that is useful for future industrial references. The Macros recording function allow the repetitive process of drawing a model to be done without the needs to draw the same model over again and again.

The important aspect that need to be considered in making the automation process possibly done is the method of parameter identification of the model. This is the key item to make the link between the coding of the form and the coding of the model to interact to each other. Moreover, the visual basic tools such as *Userform* and inputbox allow human to interact with the model by put in the input data into form and a model which is the result will be created.

By applying this knowledge, automation process in modelling a design can be done and can be fully utilized in the industrial world especially in mass production manufacturing field.

5.3 RECOMMENDATION

Firstly, in order to provide a continuous improvement in this project, the coding for macros program can be improvise to make the program produce one form only for the interface. Thus all the information for the design and command that need to be filled by the user all in one form which will make the process easier. The safety command such as *if* command can also be added into the coding if the user forgot to fill in the input data for the model, a warning message will pop out to remind the user rather than an error occured to create the model. In addition, the command can be improvised to create a multiple part in the same time.

Last but not least, most customer now demand a special editor tools in terms to purcahse a product they desired. Thus this project or program can be used as the platform for that purpose. This program can produce a more systematic ordering techniques for the customer and satisfied them by showing the detail and the result of ilustration of their product. After the customer has decided the design by using this program, then the prototype can be produced through the application such as rapid prototype manufacturing.

REFFERENCES

Autodesk, K. M. (2008). Introduction to Macro Writing, (April 2007), 1–27.

Bernard, F. (2003). A short history of CATIA & Dassault Systemes. *Aviation*, (May).

Bilalis, D. N. (2000). Computer Aided Design - CAD, 0–26.

Chandrasegaran, S. K., Ramani, K., Sriram, R. D., Horváth, I., Bernard, A., Harik, R. F., & Gao, W. (2013). The evolution, challenges, and future of knowledge representation in product design systems. *Computer-Aided Design*, *45*(2), 204–228.

Duke, B. (2014). CATIA V5 Administration Customizing Standards. *White Paper*.

Lowe, A. G., & Hartman, N. W. (2000). A Case Study in CAD Design Automation, 2–9.

Ramly, Al, E. (2013). Automated a complex computer aided design concept generated using macros programming, *12041*.

Ross, E. (n.d.). VB Scripting for CATIA V5 : How to Model Gears with VBA User Form Alireza Reihani.

Siddesh, S., & Suresh, B. S. (2015). Automation of Generating CAD Models, *5*, 55–58.

Tylee, L. (1998). Learn Visual Basic 6 . 0, *98008*(206).

Venkatesan, S., & Karnan, M. (2009). Industrial Automation. *Articlesbase.com*, (May), 1–36

Wayzode, N. D. (2012). Customization of Catia V5 for Design of Shaft Coupling. *International Conference on Emerging Frontiers in Technology for Rural Area (EFITRA)*, (L), 30–33.

Weisenberger, N. (n.d.). VB Scripting.

Yip-Hoi, D. (2010). Strategies for teaching CAD automation to engineers and technologists. *ASEE Annual Conference and Exposition, Conference Proceedings*.

Shah D. B., " Parametric Modeling and Drawing Automation for Flange Coupling using Excel Spreadsheet", International Journal of Research in Engineering and Technology, Impact Journal, 2013, p 187-192.

Cukovics S., Decedzic G., Ghionea I., "Automatic Determination of Grinding Tool Profile for Helical Surfaces Machining using CATIA/VB Interface", U.P.B.series D, 2010.

APPENDICES

Coding script for userform 'RIM DESIGN"

```
Private Sub CommandButton1_Click()
Unload UserForm1
UserForm2.Show
End Sub
Private Sub CommandButton2_Click()
Unload UserForm1
UserForm3.Show
End Sub
Private Sub CommandButton3_Click()
Unload UserForm1
UserForm4.Show
End Sub
Private Sub CommandButton4_Click()
Unload UserForm1
End Sub
```

Coding script for userform Design 1

```
Private Sub CommandButton1_Click()
Unload UserForm2
UserForm1.Show
End Sub
Private Sub enterBtn_Click()
Unload UserForm2
Dim documents1 As Documents
Set documents1 = CATIA.Documents
Dim partDocument1 As PartDocument
Set partDocument1 = documents1.Add("Part")
Dim part1 As Part
Set part1 = partDocument1.Part
Dim bodies1 As Bodies
Set bodies1 = part1.Bodies
Dim body1 As Body
Set body1 = bodies1.Item("PartBody")
Dim sketches1 As Sketches
```

```
Set sketches1 = body1.Sketches
Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements
Dim reference1 As Reference
Set reference1 = originElements1.PlaneYZ
Dim sketch1 As Sketch
Set sketch1 = sketches1.Add(reference1)
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = 0#
arrayOfVariantOfDouble1(3) = 0#
arrayOfVariantOfDouble1(4) = 1#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 0#
arrayOfVariantOfDouble1(8) = 1#
Set sketch1Variant = sketch1
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
line2D1.ReportName = 1
Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")
line2D2.ReportName = 2
Dim circle2D1 As Circle2D
Set circle2D1 = factory2D1.CreateClosedCircle(0#, 0#, 431.8)
Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")
circle2D1.CenterPoint = point2D1
circle2D1.ReportName = 3
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
```

```
Set point2D1 = axis2D1.GetItem("Origin")
circle2D1.CenterPoint = point2D1
circle2D1.ReportName = 3
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
Dim reference2 As Reference
Set reference2 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeRadius, reference2)
constraint1.Mode = catCstModeDrivingDimension
Dim length1 As Length
Set length1 = constraint1.Dimension
length1.Value = TextBox1
Dim circle2D2 As Circle2D
Set circle2D2 = factory2D1.CreateClosedCircle(0#, 0#, 406.4)
circle2D2.CenterPoint = point2D1
circle2D2.ReportName = 4
Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(406.4, 0#)
point2D2.ReportName = 5
Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(circle2D2)
Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(point2D2)
Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeOn, reference3, reference4)
constraint2.Mode = catCstModeDrivingDimension
Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(point2D2)
Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)
Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeOn, reference5, reference6)
constraint3.Mode = catCstModeDrivingDimension
Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(circle2D2)
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeDistance, reference7, reference8)
```

The value of the diameter is changed to the textbox 1 command input from the userform

```
constraint4.Mode = catCstModeDrivingDimension
Dim length2 As Length
Set length2 = constraint4.Dimension
length2.Value = 25.4
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory
Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, 20#)
pad1.IsSymmetric = True
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
Dim length3 As Length
Set length3 = limit1.Dimension
length3.Value = TextBox2 / 2
part1.UpdateObject pad1
part1.Update
Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.1;2)
;None:();Cf11:());Pad.1_ResultOUT;Z0;G4074)")
Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference9)
Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = 152.4
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2
sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2
Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()
Dim geometricElements2 As GeometricElements
```

The value of the diameter is changed to the textbox 2 command input from the userform

```
line2D3.ReportName = 1
Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")
line2D4.ReportName = 2
Dim circle2D3 As Circle2D
Set circle2D3 = factory2D2.CreateClosedCircle(0#, 0#, 406.4)
Dim point2D3 As Point2D
Set point2D3 = axis2D2.GetItem("Origin")
circle2D3.CenterPoint = point2D3
circle2D3.ReportName = 3
Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.1;2)
;None:();Cfl1:());Face:(Brp:(Pad.1;0:(Brp:(Sketch.1;4)));None:();Cfl1:());None:(Limits1:();Limits2:());Cf
11:());WithPermanentBody;WithoutBuildError;WithInitialFeatureSupport;MonoFond;MFBRepVersion_CX
R15)", pad1)
Dim geometricElements3 As GeometricElements
Set geometricElements3 = factory2D2.CreateProjections(reference10)
Dim geometry2D1 As Geometry2D
Set geometry2D1 = geometricElements3.Item("Mark.1")
geometry2D1.Construction = True
Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints
Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(circle2D3)
Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(geometry2D1)
Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeOn, reference11, reference12)
constraint5.Mode = catCstModeDrivingDimension
Dim circle2D4 As Circle2D
Set circle2D4 = factory2D2.CreateClosedCircle(0#, 0#, 457.2)
circle2D4.CenterPoint = point2D3
circle2D4.ReportName = 4
Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(457.2, 0#)
point2D4.ReportName = 5
Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(circle2D4)
Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(point2D4)
```

```
Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeOn, reference13, reference14)
constraint6.Mode = catCstModeDrivingDimension
Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(point2D4)
Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D3)
Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeOn, reference15, reference16)
constraint7.Mode = catCstModeDrivingDimension
Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(circle2D4)
Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(circle2D3)
Dim constraint8 As Constraint
Set constraint8 = constraints2.AddBiEltCst(catCstTypeDistance, reference17, reference18)
constraint8.Mode = catCstModeDrivingDimension
Dim length4 As Length
Set length4 = constraint8.Dimension
sketch2.CloseEdition
part1.InWorkObject = sketch2
part1.Update
length4.Value = 50.8
Dim pad2 As Pad
Set pad2 = shapeFactory1.AddNewPad(sketch2, 152.4)
Dim limit2 As Limit
Set limit2 = pad2.FirstLimit
Dim length5 As Length
Set length5 = limit2.Dimension
length5.Value = 12.7
part1.Update
Dim hybridShapeFactory1 As HybridShapeFactory
Set hybridShapeFactory1 = part1.HybridShapeFactory
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ
Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference19, 114.3, False)
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
Dim hybridShapes1 As HybridShapes
Set hybridShapes1 = body1.HybridShapes
Dim reference20 As Reference
Set reference20 = hybridShapes1.Item("Plane.1")
Dim sketch3 As Sketch
Set sketch3 = sketches1.Add(reference20)
Dim arrayOfVariantOfDouble3(8)
arrayOfVariantOfDouble3(0) = 114.3
arrayOfVariantOfDouble3(1) = 0#
arrayOfVariantOfDouble3(2) = 0#
arrayOfVariantOfDouble3(3) = 0#
arrayOfVariantOfDouble3(4) = 1#
arrayOfVariantOfDouble3(5) = 0#
arrayOfVariantOfDouble3(6) = 0#
arrayOfVariantOfDouble3(7) = 0#
arrayOfVariantOfDouble3(8) = 1#
Set sketch3Variant = sketch3
sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3
part1.InWorkObject = sketch3
Dim factory2D3 As Factory2D
Set factory2D3 = sketch3.OpenEdition()
Dim geometricElements4 As GeometricElements
Set geometricElements4 = sketch3.GeometricElements
Dim axis2D3 As Axis2D
Set axis2D3 = geometricElements4.Item("AbsoluteAxis")
Dim line2D5 As Line2D
Set line2D5 = axis2D3.GetItem("HDirection")
line2D5.ReportName = 1
Dim line2D6 As Line2D
Set line2D6 = axis2D3.GetItem("VDirection")
line2D6.ReportName = 2
Dim circle2D5 As Circle2D
Set circle2D5 = factory2D3.CreateClosedCircle(0#, 0#, 114.3)
Dim point2D5 As Point2D
Set point2D5 = axis2D3.GetItem("Origin")
circle2D5.CenterPoint = point2D5
circle2D5.ReportName = 3
Dim constraints3 As Constraints
```

```
Set constraints3 = sketch3.Constraints
Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(circle2D5)
Dim constraint9 As Constraint
Set constraint9 = constraints3.AddMonoEltCst(catCstTypeRadius, reference21)
constraint9.Mode = catCstModeDrivingDimension
Dim length6 As Length
Set length6 = constraint9.Dimension
length6.Value = 114.3
sketch3.CloseEdition
part1.InWorkObject = sketch3
part1.Update
Dim pad3 As Pad
Set pad3 = shapeFactory1.AddNewPad(sketch3, 12.7)
Dim limit3 As Limit
Set limit3 = pad3.FirstLimit
Dim length7 As Length
Set length7 = limit3.Dimension
length7.Value = 25.4
part1.Update
Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());Pad.3_ResultOUT;Z0;G4074)")
Dim sketch4 As Sketch
Set sketch4 = sketches1.Add(reference22)
Dim arrayOfVariantOfDouble4(8)
arrayOfVariantOfDouble4(0) = 139.7
arrayOfVariantOfDouble4(1) = 0#
arrayOfVariantOfDouble4(2) = 0#
arrayOfVariantOfDouble4(3) = 0#
arrayOfVariantOfDouble4(4) = 1#
arrayOfVariantOfDouble4(5) = 0#
arrayOfVariantOfDouble4(6) = 0#
arrayOfVariantOfDouble4(7) = 0#
arrayOfVariantOfDouble4(8) = 1#
Set sketch4Variant = sketch4
sketch4Variant.SetAbsoluteAxisData arrayOfVariantOfDouble4
part1.InWorkObject = sketch4
Dim factory2D4 As Factory2D
Set factory2D4 = sketch4.OpenEdition()
```

```
Dim geometricElements5 As GeometricElements
Set geometricElements5 = sketch4.GeometricElements
Dim axis2D4 As Axis2D
Set axis2D4 = geometricElements5.Item("AbsoluteAxis")
Dim line2D7 As Line2D
Set line2D7 = axis2D4.GetItem("HDirection")
line2D7.ReportName = 1
Dim line2D8 As Line2D
Set line2D8 = axis2D4.GetItem("VDirection")
line2D8.ReportName = 2
Dim point2D6 As Point2D
Set point2D6 = factory2D4.CreatePoint(-50.8, 403.2125)
point2D6.ReportName = 3
Dim point2D7 As Point2D
Set point2D7 = factory2D4.CreatePoint(-50.8, 102.390673)
point2D7.ReportName = 4
Dim line2D9 As Line2D
Set line2D9 = factory2D4.CreateLine(-50.8, 403.2125, -50.8, 102.390673)
line2D9.ReportName = 5
line2D9.StartPoint = point2D6
line2D9.EndPoint = point2D7
Dim constraints4 As Constraints
Set constraints4 = sketch4.Constraints
Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D9)
Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D8)
Dim constraint10 As Constraint
Set constraint10 = constraints4.AddBiEltCst(catCstTypeVerticality, reference23, reference24)
constraint10.Mode = catCstModeDrivingDimension
Dim point2D8 As Point2D
Set point2D8 = factory2D4.CreatePoint(50.8, 102.390673)
point2D8.ReportName = 6
Dim line2D10 As Line2D
Set line2D10 = factory2D4.CreateLine(-50.8, 102.390673, 50.8, 102.390673)
line2D10.ReportName = 7
line2D10.StartPoint = point2D7
line2D10.EndPoint = point2D8
Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(line2D10)
```

```
Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D7)
Dim constraint11 As Constraint
Set constraint11 = constraints4.AddBiEltCst(catCstTypeHorizontality, reference25, reference26)
constraint11.Mode = catCstModeDrivingDimension
Dim point2D9 As Point2D
Set point2D9 = factory2D4.CreatePoint(50.8, 403.2125)
point2D9.ReportName = 8
Dim line2D11 As Line2D
Set line2D11 = factory2D4.CreateLine(50.8, 102.390673, 50.8, 403.2125)
line2D11.ReportName = 9
line2D11.StartPoint = point2D8
line2D11.EndPoint = point2D9
Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D11)
Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D8)
Dim constraint12 As Constraint
Set constraint12 = constraints4.AddBiEltCst(catCstTypeVerticality, reference27, reference28)
constraint12.Mode = catCstModeDrivingDimension
Dim point2D10 As Point2D
Set point2D10 = factory2D4.CreatePoint(0#, 0#)
point2D10.ReportName = 10
Dim circle2D6 As Circle2D
Set circle2D6 = factory2D4.CreateCircle(0#, 0#, 406.4, 1.445468, 1.696124)
circle2D6.CenterPoint = point2D10
circle2D6.ReportName = 11
circle2D6.StartPoint = point2D9
circle2D6.EndPoint = point2D6
Dim point2D11 As Point2D
Set point2D11 = factory2D4.CreatePoint(0#, 406.4)
point2D11.ReportName = 12
Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(circle2D6)
Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(point2D11)
Dim constraint13 As Constraint
Set constraint13 = constraints4.AddBiEltCst(catCstTypeOn, reference29, reference30)
constraint13.Mode = catCstModeDrivingDimension
Dim reference31 As Reference
```

```
Set reference31 = part1.CreateReferenceFromObject(point2D11)
Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D8)
Dim constraint14 As Constraint
Set constraint14 = constraints4.AddBiEltCst(catCstTypeOn, reference31, reference32)
constraint14.Mode = catCstModeDrivingDimension
Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D11)
Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromObject(line2D9)
Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(line2D8)
Dim constraint15 As Constraint
Set constraint15 = constraints4.AddTriEltCst(catCstTypeSymmetry, reference34, reference35)
constraint15.Mode = catCstModeDrivingDimension
Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(line2D10)
Dim constraint16 As Constraint
Set constraint16 = constraints4.AddMonoEltCst(catCstTypeLength, reference36)
constraint16.Mode = catCstModeDrivingDimension
Dim length8 As Length
Set length8 = constraint16.Dimension
length8.Value = 101.6
Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.2;2)
;None:();Cf11:());Face:(Brp:((Brp:(Pad.2;0:(Brp:(Sketch.2;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4))))));None:()
;Cf11:());None:(Limits1:();Limits2:());Cf11:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad2)
Dim geometricElements6 As GeometricElements
Set geometricElements6 = factory2D4.CreateProjections(reference37)
Dim geometry2D2 As Geometry2D
Set geometry2D2 = geometricElements6.Item("Mark.1")
geometry2D2.Construction = True
Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(point2D9)
Dim reference39 As Reference
Set reference39 = part1.CreateReferenceFromObject(geometry2D2)
Dim constraint17 As Constraint
Set constraint17 = constraints4.AddBiEltCst(catCstTypeOn, reference38, reference39)
constraint17.Mode = catCstModeDrivingDimension
```

```
    Dim reference40 As Reference
    Set reference40 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());Face:(Brp:((Brp:(Pad.2;0:(Brp:(Sketch.2;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cfl1:());None:(Limits1:();Limits2:());Cfl1:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad2)
    Dim geometricElements7 As GeometricElements
    Set geometricElements7 = factory2D4.CreateProjections(reference40)
    Dim geometry2D3 As Geometry2D
    Set geometry2D3 = geometricElements7.Item("Mark.1")
    geometry2D3.Construction = True
    Dim reference41 As Reference
    Set reference41 = part1.CreateReferenceFromObject(point2D6)
    Dim reference42 As Reference
    Set reference42 = part1.CreateReferenceFromObject(geometry2D3)
    Dim constraint18 As Constraint
    Set constraint18 = constraints4.AddBiEltCst(catCstTypeOn, reference41, reference42)
    constraint18.Mode = catCstModeDrivingDimension
    Dim reference43 As Reference
    Set reference43 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());Face:(Brp:((Brp:(Pad.2;0:(Brp:(Sketch.2;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cfl1:());None:(Limits1:();Limits2:());Cfl1:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad2)
    Dim geometricElements8 As GeometricElements
    Set geometricElements8 = factory2D4.CreateProjections(reference43)
    Dim geometry2D4 As Geometry2D
    Set geometry2D4 = geometricElements8.Item("Mark.1")
    geometry2D4.Construction = True
    Dim reference44 As Reference
    Set reference44 = part1.CreateReferenceFromObject(point2D11)
    Dim reference45 As Reference
    Set reference45 = part1.CreateReferenceFromObject(geometry2D4)
    Dim constraint19 As Constraint
    Set constraint19 = constraints4.AddBiEltCst(catCstTypeDistance, reference44, reference45)
    constraint19.Mode = catCstModeDrivingDimension
    Dim length9 As Length
    Set length9 = constraint19.Dimension
    length9.Value = 0#
    Dim reference46 As Reference
    Set reference46 =
part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;0:(Brp:(Sketch.3;3)))
```

```
;None:();Cfl1:());Face:(Brp:(Pad.3;2);None:();Cfl1:());None:(Limits1:();Limits2:());Cfl1:());WithPerman
entBody;WithoutBuildError;WithInitialFeatureSupport;MonoFond;MFBRepVersion_CXR15)", pad3)
    Dim geometricElements9 As GeometricElements
    Set geometricElements9 = factory2D4.CreateProjections(reference46)
    Dim geometry2D5 As Geometry2D
    Set geometry2D5 = geometricElements9.Item("Mark.1")
    geometry2D5.Construction = True
    Dim reference47 As Reference
    Set reference47 = part1.CreateReferenceFromObject(point2D7)
    Dim reference48 As Reference
    Set reference48 = part1.CreateReferenceFromObject(geometry2D5)
    Dim constraint20 As Constraint
    Set constraint20 = constraints4.AddBiEltCst(catCstTypeOn, reference47, reference48)
    constraint20.Mode = catCstModeDrivingDimension
    sketch4.CloseEdition
    part1.InWorkObject = sketch4
    part1.Update
    length9.Value = 0#
    Dim pad4 As Pad
    Set pad4 = shapeFactory1.AddNewPad(sketch4, 25.4)
    pad4.IsSymmetric = True
    Dim limit4 As Limit
    Set limit4 = pad4.FirstLimit
    Dim length10 As Length
    Set length10 = limit4.Dimension
    length10.Value = 15.24
    part1.Update
    Dim reference49 As Reference
    Set reference49 = part1.CreateReferenceFromName("")
    Dim reference50 As Reference
    Set reference50 = part1.CreateReferenceFromName("")
    Dim circPattern1 As CircPattern
    Set circPattern1 = shapeFactory1.AddNewCircPattern(Nothing, 1, 2, 20#, 45#, 1, 1, reference49,
reference50, True, 0#, True)
    circPattern1.CircularPatternParameters = catInstancesandAngularSpacing
    Dim angularRepartition1 As AngularRepartition
    Set angularRepartition1 = circPattern1.AngularRepartition
    Dim intParam1 As IntParam
    Set intParam1 = angularRepartition1.InstancesCount
    intParam1.Value = 5
```

```
Dim angularRepartition1 As AngularRepartition
Set angularRepartition1 = circPattern1.AngularRepartition
Dim intParam1 As IntParam
Set intParam1 = angularRepartition1.InstancesCount

Dim angularRepartition2 As AngularRepartition
Set angularRepartition2 = circPattern1.AngularRepartition
Dim angle1 As Angle
Set angle1 = angularRepartition2.AngularSpacing
angle1.Value = 360 / TextBox3
intParam1.Value = TextBox3
circPattern1.ItemToCopy = pad4
Dim reference51 As Reference
Set reference51 = part1.CreateReferenceFromBRepName("FSur:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());WithTemporaryBody;WithoutBuildError;WithInitialFeatureSupport;MFBRepVersion_CX
R15)", pad3)
circPattern1.SetRotationAxis reference51
part1.Update
Dim reference52 As Reference
Set reference52 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());CircPattern.1_ResultOUT;Z0;G4074)")
Dim sketch5 As Sketch
Set sketch5 = sketches1.Add(reference52)
Dim arrayOfVariantOfDouble5(8)
arrayOfVariantOfDouble5(0) = 139.7
arrayOfVariantOfDouble5(1) = 0#
arrayOfVariantOfDouble5(2) = 0#
arrayOfVariantOfDouble5(3) = 0#
arrayOfVariantOfDouble5(4) = 1#
arrayOfVariantOfDouble5(5) = 0#
arrayOfVariantOfDouble5(6) = 0#
arrayOfVariantOfDouble5(7) = 0#
arrayOfVariantOfDouble5(8) = 1#
Set sketch5Variant = sketch5
sketch5Variant.SetAbsoluteAxisData arrayOfVariantOfDouble5
part1.InWorkObject = sketch5
Dim factory2D5 As Factory2D
Set factory2D5 = sketch5.OpenEdition()
Dim geometricElements10 As GeometricElements
Set geometricElements10 = sketch5.GeometricElements
```

The value of the diameter is changed to the textbox 3 command input from the userform

```
Dim line2D13 As Line2D
Set line2D13 = axis2D5.GetItem("VDirection")
line2D13.ReportName = 2
Dim circle2D7 As Circle2D
Set circle2D7 = factory2D5.CreateClosedCircle(0#, 0#, 50.8)
Dim point2D12 As Point2D
Set point2D12 = axis2D5.GetItem("Origin")
circle2D7.CenterPoint = point2D12
circle2D7.ReportName = 3
Dim constraints5 As Constraints
Set constraints5 = sketch5.Constraints
Dim reference53 As Reference
Set reference53 = part1.CreateReferenceFromObject(circle2D7)
Dim constraint21 As Constraint
Set constraint21 = constraints5.AddMonoEltCst(catCstTypeRadius, reference53)
constraint21.Mode = catCstModeDrivingDimension
Dim length11 As Length
Set length11 = constraint21.Dimension
length11.Value = 50.8
sketch5.CloseEdition
part1.InWorkObject = sketch5
part1.Update
Dim pad5 As Pad
Set pad5 = shapeFactory1.AddNewPad(sketch5, 15.24)
Dim limit5 As Limit
Set limit5 = pad5.FirstLimit
Dim length12 As Length
Set length12 = limit5.Dimension
length12.Value = 10.16
part1.Update
Dim reference54 As Reference
Set reference54 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.3;2)
;None:();Cfl1:());Pad.5_ResultOUT;Z0;G4074)")
Dim sketch6 As Sketch
Set sketch6 = sketches1.Add(reference54)
Dim arrayOfVariantOfDouble6(8)
arrayOfVariantOfDouble6(0) = 139.7
arrayOfVariantOfDouble6(1) = 0#
arrayOfVariantOfDouble6(2) = 0#
arrayOfVariantOfDouble6(3) = 0#
```

```
arrayOfVariantOfDouble6(4) = 1#
arrayOfVariantOfDouble6(5) = 0#
arrayOfVariantOfDouble6(6) = 0#
arrayOfVariantOfDouble6(7) = 0#
arrayOfVariantOfDouble6(8) = 1#
Set sketch6Variant = sketch6
sketch6Variant.SetAbsoluteAxisData arrayOfVariantOfDouble6
part1.InWorkObject = sketch6
Dim factory2D6 As Factory2D
Set factory2D6 = sketch6.OpenEdition()
Dim geometricElements11 As GeometricElements
Set geometricElements11 = sketch6.GeometricElements
Dim axis2D6 As Axis2D
Set axis2D6 = geometricElements11.Item("AbsoluteAxis")
Dim line2D14 As Line2D
Set line2D14 = axis2D6.GetItem("HDirection")
line2D14.ReportName = 1
Dim line2D15 As Line2D
Set line2D15 = axis2D6.GetItem("VDirection")
line2D15.ReportName = 2
Dim point2D13 As Point2D
Set point2D13 = factory2D6.CreatePoint(0#, 76.2)
point2D13.ReportName = 3
Dim circle2D8 As Circle2D
Set circle2D8 = factory2D6.CreateClosedCircle(0#, 76.2, 12.7)
circle2D8.CenterPoint = point2D13
circle2D8.ReportName = 4
Dim constraints6 As Constraints
Set constraints6 = sketch6.Constraints
Dim reference55 As Reference
Set reference55 = part1.CreateReferenceFromObject(point2D13)
Dim reference56 As Reference
Set reference56 = part1.CreateReferenceFromObject(line2D15)
Dim constraint22 As Constraint
Set constraint22 = constraints6.AddBiEltCst(catCstTypeOn, reference55, reference56)
constraint22.Mode = catCstModeDrivingDimension
Dim reference57 As Reference
Set reference57 = part1.CreateReferenceFromObject(circle2D8)
Dim constraint23 As Constraint
Set constraint23 = constraints6.AddMonoEltCst(catCstTypeRadius, reference57)
```

```
constraint23.Mode = catCstModeDrivingDimension
Dim length13 As Length
Set length13 = constraint23.Dimension
length13.Value = 12.7
Dim reference58 As Reference
Set reference58 = part1.CreateReferenceFromObject(point2D13)
Dim point2D14 As Point2D
Set point2D14 = axis2D6.GetItem("Origin")
Dim reference59 As Reference
Set reference59 = part1.CreateReferenceFromObject(point2D14)
Dim constraint24 As Constraint
Set constraint24 = constraints6.AddBiEltCst(catCstTypeDistance, reference58, reference59)
constraint24.Mode = catCstModeDrivingDimension
Dim length14 As Length
Set length14 = constraint24.Dimension
length14.Value = 76.2
sketch6.CloseEdition
part1.InWorkObject = sketch6
part1.Update
Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch6, 10.16)
Dim limit6 As Limit
Set limit6 = pocket1.FirstLimit
limit6.LimitMode = catUpToLastLimit
part1.UpdateObject pocket1
part1.Update
Dim reference60 As Reference
Set reference60 = part1.CreateReferenceFromName("")
Dim reference61 As Reference
Set reference61 = part1.CreateReferenceFromName("")
Dim circPattern2 As CircPattern
Set circPattern2 = shapeFactory1.AddNewCircPattern(Nothing, 1, 2, 20#, 45#, 1, 1, reference60,
reference61, True, 0#, True)
circPattern2.CircularPatternParameters = catInstancesandAngularSpacing
Dim angularRepartition3 As AngularRepartition
Set angularRepartition3 = circPattern2.AngularRepartition
Dim intParam2 As IntParam
Set intParam2 = angularRepartition3.InstancesCount
intParam2.Value = 4
Dim angularRepartition4 As AngularRepartition
```

```
Set angularRepartition3 = circPattern2.AngularRepartition
Dim intParam2 As IntParam
Set intParam2 = angularRepartition3.InstancesCount
intParam2.Value = 4
Dim angularRepartition4 As AngularRepartition
Set angularRepartition4 = circPattern2.AngularRepartition
Dim angle2 As Angle
Set angle2 = angularRepartition4.AngularSpacing
angle2.Value = 360 / TextBox4
intParam2.Value = TextBox4
circPattern2.ItemToCopy = pocket1
Dim reference62 As Reference
Set reference62 = part1.CreateReferenceFromBRepName("FSur:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());WithTemporaryBody;WithoutBuildError;WithInitialFeatureSupport;MFBRepVersion_CX
R15)", pad3)
circPattern2.SetRotationAxis reference62
part1.Update
End Sub
```

The value of the diameter is changed to the textbox 4 command input from the userform

Coding script for userfrom Design2

```
Private Sub CommandButton1_Click()
Unload UserForm3
Dim documents1 As Documents
Set documents1 = CATIA.Documents
Dim partDocument1 As PartDocument
Set partDocument1 = documents1.Add("Part")
Dim part1 As Part
Set part1 = partDocument1.Part
Dim bodies1 As Bodies
Set bodies1 = part1.Bodies
Dim body1 As Body
Set body1 = bodies1.Item("PartBody")
Dim sketches1 As Sketches
Set sketches1 = body1.Sketches
Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements
Dim reference1 As Reference
Set reference1 = originElements1.PlaneYZ
Dim sketch1 As Sketch
Set sketch1 = sketches1.Add(reference1)
```

```
arrayOfVariantOfDouble1(4) = 1#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 0#
arrayOfVariantOfDouble1(8) = 1#
Set sketch1Variant = sketch1
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
line2D1.ReportName = 1
Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")
line2D2.ReportName = 2
Dim circle2D1 As Circle2D
Set circle2D1 = factory2D1.CreateClosedCircle(0#, 0#, 431.8)
Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")
circle2D1.CenterPoint = point2D1
circle2D1.ReportName = 3
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
Dim reference2 As Reference
Set reference2 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeRadius, reference2)
constraint1.Mode = catCstModeDrivingDimension
Dim length1 As Length
Set length1 = constraint1.Dimension
length1.Value = TextBox1
Dim circle2D2 As Circle2D
Set circle2D2 = factory2D1.CreateClosedCircle(0#, 0#, 393.7)
circle2D2.CenterPoint = point2D1
circle2D2.ReportName = 4
```

```
Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(393.7, 0#)
point2D2.ReportName = 5
Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(circle2D2)
Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(point2D2)
Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeOn, reference3, reference4)
constraint2.Mode = catCstModeDrivingDimension
Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(point2D2)
Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)
Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeOn, reference5, reference6)
constraint3.Mode = catCstModeDrivingDimension
Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(circle2D2)
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeDistance, reference7, reference8)
constraint4.Mode = catCstModeDrivingDimension
Dim length2 As Length
Set length2 = constraint4.Dimension
length2.Value = TextBox2 - 40
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
Set factory2D1 = sketch1.OpenEdition()
circle2D2.SetData 0#, 0#, 406.4
point2D2.SetData 406.4, 0#
length2.Value = 25.4
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory
Dim pad1 As Pad
```

```
Set pad1 = shapeFactory1.AddNewPad(sketch1, 20#)
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
Dim length3 As Length
Set length3 = limit1.Dimension
length3.Value = TextBox2 / 2
pad1.IsSymmetric = True
part1.Update
Dim hybridShapeFactory1 As HybridShapeFactory
Set hybridShapeFactory1 = part1.HybridShapeFactory
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ
Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference9, TextBox2 / 2 -
50, False)
body1.InsertHybridShape hybridShapePlaneOffset1
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
Dim hybridShapes1 As HybridShapes
Set hybridShapes1 = body1.HybridShapes
Dim reference10 As Reference
Set reference10 = hybridShapes1.Item("Plane.1")
Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference10)
Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = 114.3
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2
sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2
Dim factory2D2 As Factory2D
```

```
Set factory2D2 = sketch2.OpenEdition()
Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements
Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")
line2D3.ReportName = 1
Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")
line2D4.ReportName = 2
Dim circle2D3 As Circle2D
Set circle2D3 = factory2D2.CreateClosedCircle(0#, 0#, 88.9)
Dim point2D3 As Point2D
Set point2D3 = axis2D2.GetItem("Origin")
circle2D3.CenterPoint = point2D3
circle2D3.ReportName = 3
Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(88.9, 0#)
point2D4.ReportName = 4
Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints
Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(circle2D3)
Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(point2D4)
Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeOn, reference11, reference12)
constraint5.Mode = catCstModeDrivingDimension
Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(point2D4)
Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)
Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeOn, reference13, reference14)
constraint6.Mode = catCstModeDrivingDimension
Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(circle2D3)
Dim constraint7 As Constraint
Set constraint7 = constraints2.AddMonoEltCst(catCstTypeRadius, reference15)
```

```
constraint7.Mode = catCstModeDrivingDimension
Dim length4 As Length
Set length4 = constraint7.Dimension
length4.Value = 88.9
sketch2.CloseEdition
part1.InWorkObject = sketch2
part1.Update
Dim pad2 As Pad
Set pad2 = shapeFactory1.AddNewPad(sketch2, 152.4)
Dim limit2 As Limit
Set limit2 = pad2.FirstLimit
Dim length5 As Length
Set length5 = limit2.Dimension
length5.Value = 25.4
part1.Update
Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.1;2)
;None:();Cfl1:());Pad.2_ResultOUT;Z0;G4074)")
Dim sketch3 As Sketch
Set sketch3 = sketches1.Add(reference16)
Dim arrayOfVariantOfDouble3(8)
arrayOfVariantOfDouble3(0) = 152.4
arrayOfVariantOfDouble3(1) = 0#
arrayOfVariantOfDouble3(2) = 0#
arrayOfVariantOfDouble3(3) = 0#
arrayOfVariantOfDouble3(4) = 1#
arrayOfVariantOfDouble3(5) = 0#
arrayOfVariantOfDouble3(6) = 0#
arrayOfVariantOfDouble3(7) = 0#
arrayOfVariantOfDouble3(8) = 1#
Set sketch3Variant = sketch3
sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3
part1.InWorkObject = sketch3
Dim factory2D3 As Factory2D
Set factory2D3 = sketch3.OpenEdition()
Dim geometricElements3 As GeometricElements
Set geometricElements3 = sketch3.GeometricElements
Dim axis2D3 As Axis2D
Set axis2D3 = geometricElements3.Item("AbsoluteAxis")
Dim line2D5 As Line2D
```

```
Set line2D5 = axis2D3.GetItem("HDirection")
line2D5.ReportName = 1
Dim line2D6 As Line2D
Set line2D6 = axis2D3.GetItem("VDirection")
line2D6.ReportName = 2
Dim circle2D4 As Circle2D
Set circle2D4 = factory2D3.CreateClosedCircle(0#, 0#, 406.4)
Dim point2D5 As Point2D
Set point2D5 = axis2D3.GetItem("Origin")
circle2D4.CenterPoint = point2D5
circle2D4.ReportName = 3
Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.1;2)
;None:();Cf11:());Face:(Brp:(Pad.1;0:(Brp:(Sketch.1;4)));None:();Cf11:());None:(Limits1:();Limits2:());Cf
11:());WithPermanentBody;WithoutBuildError;WithInitialFeatureSupport;MonoFond;MFBRepVersion_CX
R15)", pad1)
Dim geometricElements4 As GeometricElements
Set geometricElements4 = factory2D3.CreateProjections(reference17)
Dim geometry2D1 As Geometry2D
Set geometry2D1 = geometricElements4.Item("Mark.1")
geometry2D1.Construction = True
Dim constraints3 As Constraints
Set constraints3 = sketch3.Constraints
Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(circle2D4)
Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(geometry2D1)
Dim constraint8 As Constraint
Set constraint8 = constraints3.AddBiEltCst(catCstTypeOn, reference18, reference19)
constraint8.Mode = catCstModeDrivingDimension
Dim circle2D5 As Circle2D
Set circle2D5 = factory2D3.CreateClosedCircle(0#, 0#, 444.5)
circle2D5.CenterPoint = point2D5
circle2D5.ReportName = 4
Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(circle2D5)
Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(circle2D4)
Dim constraint9 As Constraint
Set constraint9 = constraints3.AddBiEltCst(catCstTypeDistance, reference20, reference21)
```

```
constraint9.Mode = catCstModeDrivingDimension
Dim length6 As Length
Set length6 = constraint9.Dimension
length6.Value = 38.1
sketch3.CloseEdition
part1.InWorkObject = sketch3
part1.Update
length6.Value = 38.1
Dim pad3 As Pad
Set pad3 = shapeFactory1.AddNewPad(sketch3, 25.4)
part1.Update
Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.2;2)
;None:();Cf11:());Pad.3_ResultOUT;Z0;G4074)")
Dim sketch4 As Sketch
Set sketch4 = sketches1.Add(reference22)
Dim arrayOfVariantOfDouble4(8)
arrayOfVariantOfDouble4(0) = 139.7
arrayOfVariantOfDouble4(1) = 0#
arrayOfVariantOfDouble4(2) = 0#
arrayOfVariantOfDouble4(3) = 0#
arrayOfVariantOfDouble4(4) = 1#
arrayOfVariantOfDouble4(5) = 0#
arrayOfVariantOfDouble4(6) = 0#
arrayOfVariantOfDouble4(7) = 0#
arrayOfVariantOfDouble4(8) = 1#
Set sketch4Variant = sketch4
sketch4Variant.SetAbsoluteAxisData arrayOfVariantOfDouble4
part1.InWorkObject = sketch4
Dim factory2D4 As Factory2D
Set factory2D4 = sketch4.OpenEdition()
Dim geometricElements5 As GeometricElements
Set geometricElements5 = sketch4.GeometricElements
Dim axis2D4 As Axis2D
Set axis2D4 = geometricElements5.Item("AbsoluteAxis")
Dim line2D7 As Line2D
Set line2D7 = axis2D4.GetItem("HDirection")
line2D7.ReportName = 1
Dim line2D8 As Line2D
Set line2D8 = axis2D4.GetItem("VDirection")
```

```
line2D8.ReportName = 2
Dim point2D6 As Point2D
Set point2D6 = factory2D4.CreatePoint(-31.75, 83.037025)
point2D6.ReportName = 3
Dim point2D7 As Point2D
Set point2D7 = factory2D4.CreatePoint(-31.75, 210.037025)
point2D7.ReportName = 4
Dim line2D9 As Line2D
Set line2D9 = factory2D4.CreateLine(-31.75, 83.037025, -31.75, 210.037025)
line2D9.ReportName = 5
line2D9.StartPoint = point2D6
line2D9.EndPoint = point2D7
Dim constraints4 As Constraints
Set constraints4 = sketch4.Constraints
Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D9)
Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D8)
Dim constraint10 As Constraint
Set constraint10 = constraints4.AddBiEltCst(catCstTypeVerticality, reference23, reference24)
constraint10.Mode = catCstModeDrivingDimension
Dim point2D8 As Point2D
Set point2D8 = factory2D4.CreatePoint(-128.145145, 385.667968)
point2D8.ReportName = 6
Dim line2D10 As Line2D
Set line2D10 = factory2D4.CreateLine(-31.75, 210.037025, -128.145145, 385.667968)
line2D10.ReportName = 7
line2D10.StartPoint = point2D7
line2D10.EndPoint = point2D8
Dim point2D9 As Point2D
Set point2D9 = factory2D4.CreatePoint(-97.30347, 368.365355)
point2D9.ReportName = 8
Dim point2D10 As Point2D
Set point2D10 = factory2D4.CreatePoint(-79.029764, 398.641764)
point2D10.ReportName = 9
Dim circle2D6 As Circle2D
Set circle2D6 = factory2D4.CreateCircle(-97.30347, 368.365355, 35.363672, 1.027761, 2.630333)
circle2D6.CenterPoint = point2D9
circle2D6.ReportName = 10
circle2D6.EndPoint = point2D8
```

```
circle2D6.StartPoint = point2D10
Dim point2D11 As Point2D
Set point2D11 = factory2D4.CreatePoint(0#, 319.612)
point2D11.ReportName = 11
Dim line2D11 As Line2D
Set line2D11 = factory2D4.CreateLine(-79.029764, 398.641764, 0#, 319.612)
line2D11.ReportName = 12
line2D11.StartPoint = point2D10
line2D11.EndPoint = point2D11
Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(point2D11)
Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D8)
Dim constraint11 As Constraint
Set constraint11 = constraints4.AddBiEltCst(catCstTypeOn, reference25, reference26)
constraint11.Mode = catCstModeDrivingDimension
Dim point2D12 As Point2D
Set point2D12 = factory2D4.CreatePoint(79.029764, 398.641764)
point2D12.ReportName = 13
Dim line2D12 As Line2D
Set line2D12 = factory2D4.CreateLine(0#, 319.612, 79.029764, 398.641764)
line2D12.ReportName = 14
line2D12.StartPoint = point2D11
line2D12.EndPoint = point2D12
Dim point2D13 As Point2D
Set point2D13 = factory2D4.CreatePoint(97.30347, 368.365355)
point2D13.ReportName = 15
Dim point2D14 As Point2D
Set point2D14 = factory2D4.CreatePoint(128.145145, 385.667968)
point2D14.ReportName = 16
Dim circle2D7 As Circle2D
Set circle2D7 = factory2D4.CreateCircle(97.30347, 368.365355, 35.363672, 0.51126, 2.113831)
circle2D7.CenterPoint = point2D13
circle2D7.ReportName = 17
circle2D7.EndPoint = point2D12
circle2D7.StartPoint = point2D14
Dim point2D15 As Point2D
Set point2D15 = factory2D4.CreatePoint(31.75, 210.037025)
point2D15.ReportName = 18
Dim line2D13 As Line2D
```

```
Set line2D13 = factory2D4.CreateLine(128.145145, 385.667968, 31.75, 210.037025)
line2D13.ReportName = 19
line2D13.StartPoint = point2D14
line2D13.EndPoint = point2D15
Dim point2D16 As Point2D
Set point2D16 = factory2D4.CreatePoint(31.75, 83.037025)
point2D16.ReportName = 20
Dim line2D14 As Line2D
Set line2D14 = factory2D4.CreateLine(31.75, 210.037025, 31.75, 83.037025)
line2D14.ReportName = 21
line2D14.StartPoint = point2D15
line2D14.EndPoint = point2D16
Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D14)
Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D8)
Dim constraint12 As Constraint
Set constraint12 = constraints4.AddBiEltCst(catCstTypeVerticality, reference27, reference28)
constraint12.Mode = catCstModeDrivingDimension
Dim line2D15 As Line2D
Set line2D15 = factory2D4.CreateLine(31.75, 83.037025, -31.75, 83.037025)
line2D15.ReportName = 22
line2D15.StartPoint = point2D16
line2D15.EndPoint = point2D6
Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(line2D15)
Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(line2D7)
Dim constraint13 As Constraint
Set constraint13 = constraints4.AddBiEltCst(catCstTypeHorizontality, reference29, reference30)
constraint13.Mode = catCstModeDrivingDimension
Dim reference31 As Reference
Set reference31 = part1.CreateReferenceFromObject(line2D9)
Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D14)
Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D8)
Dim constraint14 As Constraint
Set constraint14 = constraints4.AddTriEltCst(catCstTypeSymmetry, reference31, reference32,
reference33)
```

```
constraint14.Mode = catCstModeDrivingDimension
Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.2;0:(Brp:(Sketch.2;3)));None:();Cf11:());Face:(Brp:(Pad.2;2);None:();Cf11:());None:(Limits1:();Limits2:());Cf11:());WithPermanentBody;WithoutBuildError;WithInitialFeatureSupport;MonoFond;MFBRepVersion_CXR15)", pad2)
Dim geometricElements6 As GeometricElements
Set geometricElements6 = factory2D4.CreateProjections(reference34)
Dim geometry2D2 As Geometry2D
Set geometry2D2 = geometricElements6.Item("Mark.1")
geometry2D2.Construction = True
Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(point2D6)
Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(geometry2D2)
Dim constraint15 As Constraint
Set constraint15 = constraints4.AddBiEltCst(catCstTypeOn, reference35, reference36)
constraint15.Mode = catCstModeDrivingDimension
Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromObject(line2D15)
Dim constraint16 As Constraint
Set constraint16 = constraints4.AddMonoEltCst(catCstTypeLength, reference37)
constraint16.Mode = catCstModeDrivingDimension
Dim length7 As Length
Set length7 = constraint16.Dimension
length7.Value = 63.5
Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(point2D7)
Dim reference39 As Reference
Set reference39 = part1.CreateReferenceFromObject(point2D15)
Dim reference40 As Reference
Set reference40 = part1.CreateReferenceFromObject(line2D8)
Dim constraint17 As Constraint
Set constraint17 = constraints4.AddTriEltCst(catCstTypeSymmetry, reference38, reference39, reference40)
constraint17.Mode = catCstModeDrivingDimension
Dim reference41 As Reference
Set reference41 = part1.CreateReferenceFromObject(point2D8)
Dim reference42 As Reference
Set reference42 = part1.CreateReferenceFromObject(point2D14)
```

```
    Dim reference43 As Reference
    Set reference43 = part1.CreateReferenceFromObject(line2D8)
    Dim constraint18 As Constraint
    Set constraint18 = constraints4.AddTriEltCst(catCstTypeSymmetry, reference41, reference42,
reference43)
    constraint18.Mode = catCstModeDrivingDimension
    Dim reference44 As Reference
    Set reference44 = part1.CreateReferenceFromObject(point2D12)
    Dim reference45 As Reference
    Set reference45 = part1.CreateReferenceFromObject(point2D10)
    Dim reference46 As Reference
    Set reference46 = part1.CreateReferenceFromObject(line2D8)
    Dim constraint19 As Constraint
    Set constraint19 = constraints4.AddTriEltCst(catCstTypeSymmetry, reference44, reference45,
reference46)
    constraint19.Mode = catCstModeDrivingDimension
    Dim reference47 As Reference
    Set reference47 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());Face:(Brp:((Brp:(Pad.3;0:(Brp:(Sketch.3;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cf11:());None:(Limits1:();Limits2:());Cf11:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad3)
    Dim geometricElements7 As GeometricElements
    Set geometricElements7 = factory2D4.CreateProjections(reference47)
    Dim geometry2D3 As Geometry2D
    Set geometry2D3 = geometricElements7.Item("Mark.1")
    geometry2D3.Construction = True
    Dim reference48 As Reference
    Set reference48 = part1.CreateReferenceFromObject(point2D8)
    Dim reference49 As Reference
    Set reference49 = part1.CreateReferenceFromObject(geometry2D3)
    Dim constraint20 As Constraint
    Set constraint20 = constraints4.AddBiEltCst(catCstTypeOn, reference48, reference49)
    constraint20.Mode = catCstModeDrivingDimension
    Dim reference50 As Reference
    Set reference50 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());Face:(Brp:((Brp:(Pad.3;0:(Brp:(Sketch.3;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cf11:());None:(Limits1:();Limits2:());Cf11:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad3)
    Dim geometricElements8 As GeometricElements
    Set geometricElements8 = factory2D4.CreateProjections(reference50)
```

```
Dim geometry2D4 As Geometry2D
Set geometry2D4 = geometricElements8.Item("Mark.1")
geometry2D4.Construction = True
Dim reference51 As Reference
Set reference51 = part1.CreateReferenceFromObject(point2D10)
Dim reference52 As Reference
Set reference52 = part1.CreateReferenceFromObject(geometry2D4)
Dim constraint21 As Constraint
Set constraint21 = constraints4.AddBiEltCst(catCstTypeOn, reference51, reference52)
constraint21.Mode = catCstModeDrivingDimension
Dim reference53 As Reference
Set reference53 = part1.CreateReferenceFromObject(line2D11)
Dim reference54 As Reference
Set reference54 = part1.CreateReferenceFromObject(line2D12)
Dim constraint22 As Constraint
Set constraint22 = constraints4.AddBiEltCst(catCstTypeAngle, reference53, reference54)
constraint22.Mode = catCstModeDrivingDimension
constraint22.AngleSector = catCstAngleSector2
Dim angle1 As Angle
Set angle1 = constraint22.Dimension
angle1.Value = 90#
Dim reference55 As Reference
Set reference55 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;2)
;None:();Cf11:());Face:(Brp:((Brp:(Pad.3;0:(Brp:(Sketch.3;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4))))));None:()
;Cf11:());None:(Limits1:();Limits2:());Cf11:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad3)
Dim geometricElements9 As GeometricElements
Set geometricElements9 = factory2D4.CreateProjections(reference55)
Dim geometry2D5 As Geometry2D
Set geometry2D5 = geometricElements9.Item("Mark.1")
geometry2D5.Construction = True
Dim reference56 As Reference
Set reference56 = part1.CreateReferenceFromObject(point2D13)
Dim reference57 As Reference
Set reference57 = part1.CreateReferenceFromObject(geometry2D5)
Dim constraint23 As Constraint
Set constraint23 = constraints4.AddBiEltCst(catCstTypeDistance, reference56, reference57)
constraint23.Mode = catCstModeDrivingDimension
Dim length8 As Length
Set length8 = constraint23.Dimension
```

```
length8.Value = 25.4
Dim reference58 As Reference
Set reference58 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;2)
;None:();Cfl1:());Face:(Brp:((Brp:(Pad.3;0:(Brp:(Sketch.3;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cfl1:());None:(Limits1:();Limits2:());Cfl1:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad3)
Dim geometricElements10 As GeometricElements
Set geometricElements10 = factory2D4.CreateProjections(reference58)
Dim geometry2D6 As Geometry2D
Set geometry2D6 = geometricElements10.Item("Mark.1")
geometry2D6.Construction = True
Dim reference59 As Reference
Set reference59 = part1.CreateReferenceFromObject(point2D9)
Dim reference60 As Reference
Set reference60 = part1.CreateReferenceFromObject(geometry2D6)
Dim constraint24 As Constraint
Set constraint24 = constraints4.AddBiEltCst(catCstTypeDistance, reference59, reference60)
constraint24.Mode = catCstModeDrivingDimension
Dim length9 As Length
Set length9 = constraint24.Dimension
length9.Value = 25.4
Dim reference61 As Reference
Set reference61 = part1.CreateReferenceFromObject(line2D14)
Dim constraint25 As Constraint
Set constraint25 = constraints4.AddMonoEltCst(catCstTypeLength, reference61)
constraint25.Mode = catCstModeDrivingDimension
Dim length10 As Length
Set length10 = constraint25.Dimension
length10.Value = 127#
Dim reference62 As Reference
Set reference62 = part1.CreateReferenceFromBRepName("FEdge:(Edge:(Face:(Brp:(Pad.3;2)
;None:();Cfl1:());Face:(Brp:((Brp:(Pad.3;0:(Brp:(Sketch.3;3)));Brp:(Pad.1;0:(Brp:(Sketch.1;4)))));None:()
;Cfl1:());None:(Limits1:();Limits2:());Cfl1:());WithPermanentBody;WithoutBuildError;WithInitialFeature
Support;MonoFond;MFBRepVersion_CXR15)", pad3)
Dim geometricElements11 As GeometricElements
Set geometricElements11 = factory2D4.CreateProjections(reference62)
Dim geometry2D7 As Geometry2D
Set geometry2D7 = geometricElements11.Item("Mark.1")
geometry2D7.Construction = True
Dim reference63 As Reference
```

```
Set reference63 = part1.CreateReferenceFromObject(point2D11)
Dim reference64 As Reference
Set reference64 = part1.CreateReferenceFromObject(geometry2D7)
Dim constraint26 As Constraint
Set constraint26 = constraints4.AddBiEltCst(catCstTypeDistance, reference63, reference64)
constraint26.Mode = catCstModeDrivingDimension
Dim length11 As Length
Set length11 = constraint26.Dimension
length11.Value = 86.788
Dim reference65 As Reference
Set reference65 = part1.CreateReferenceFromObject(point2D12)
Dim reference66 As Reference
Set reference66 = part1.CreateReferenceFromObject(point2D14)
Dim constraint27 As Constraint
Set constraint27 = constraints4.AddBiEltCst(catCstTypeDistance, reference65, reference66)
constraint27.Mode = catCstModeDrivingDimension
Dim length12 As Length
Set length12 = constraint27.Dimension
length12.Value = 50.8
sketch4.CloseEdition
part1.InWorkObject = sketch4
part1.Update
length8.Value = 25.4
length9.Value = 25.4
length11.Value = 86.788
length12.Value = 50.8
Dim pad4 As Pad
Set pad4 = shapeFactory1.AddNewPad(sketch4, 25.4)
pad4.IsSymmetric = True
Dim limit3 As Limit
Set limit3 = pad4.FirstLimit
Dim length13 As Length
Set length13 = limit3.Dimension
length13.Value = 12.7
part1.Update
Dim reference67 As Reference
Set reference67 = part1.CreateReferenceFromName("")
Dim reference68 As Reference
Set reference68 = part1.CreateReferenceFromName("")
Dim circPattern1 As CircPattern
```

```
    Set circPattern1 = shapeFactory1.AddNewCircPattern(pad4, 1, 2, 20#, 45#, 1, 1, reference67,
reference68, True, 0#, True)
    circPattern1.CircularPatternParameters = catInstancesandAngularSpacing
    Dim angularRepartition1 As AngularRepartition
    Set angularRepartition1 = circPattern1.AngularRepartition
    Dim intParam1 As IntParam
    Set intParam1 = angularRepartition1.InstancesCount
    Dim angularRepartition2 As AngularRepartition
    Set angularRepartition2 = circPattern1.AngularRepartition
    Dim angle2 As Angle
    Set angle2 = angularRepartition2.AngularSpacing
    angle2.Value = 360 / TextBox3
    intParam1.Value = TextBox3
    Dim reference69 As Reference
    Set reference69 = part1.CreateReferenceFromBRepName("FSur:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());WithTemporaryBody;WithoutBuildError;WithInitialFeatureSupport;MFBRepVersion_CX
R15)", pad2)
    circPattern1.SetRotationAxis reference69
    part1.Update
    Dim reference70 As Reference
    Set reference70 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());CircPattern.1_ResultOUT;Z0;G4074)")
    Dim sketch5 As Sketch
    Set sketch5 = sketches1.Add(reference70)
    Dim arrayOfVariantOfDouble5(8)
    arrayOfVariantOfDouble5(0) = 139.7
    arrayOfVariantOfDouble5(1) = 0#
    arrayOfVariantOfDouble5(2) = 0#
    arrayOfVariantOfDouble5(3) = 0#
    arrayOfVariantOfDouble5(4) = 1#
    arrayOfVariantOfDouble5(5) = 0#
    arrayOfVariantOfDouble5(6) = 0#
    arrayOfVariantOfDouble5(7) = 0#
    arrayOfVariantOfDouble5(8) = 1#
    Set sketch5Variant = sketch5
    sketch5Variant.SetAbsoluteAxisData arrayOfVariantOfDouble5
    part1.InWorkObject = sketch5
    Dim factory2D5 As Factory2D
    Set factory2D5 = sketch5.OpenEdition()
    Dim geometricElements12 As GeometricElements
```

```
Set geometricElements12 = sketch5.GeometricElements
Dim axis2D5 As Axis2D
Set axis2D5 = geometricElements12.Item("AbsoluteAxis")
Dim line2D16 As Line2D
Set line2D16 = axis2D5.GetItem("HDirection")
line2D16.ReportName = 1
Dim line2D17 As Line2D
Set line2D17 = axis2D5.GetItem("VDirection")
line2D17.ReportName = 2
Dim circle2D8 As Circle2D
Set circle2D8 = factory2D5.CreateClosedCircle(0#, 0#, 38.1)
Dim point2D17 As Point2D
Set point2D17 = axis2D5.GetItem("Origin")
circle2D8.CenterPoint = point2D17
circle2D8.ReportName = 3
Dim constraints5 As Constraints
Set constraints5 = sketch5.Constraints
Dim reference71 As Reference
Set reference71 = part1.CreateReferenceFromObject(circle2D8)
Dim constraint28 As Constraint
Set constraint28 = constraints5.AddMonoEltCst(catCstTypeRadius, reference71)
constraint28.Mode = catCstModeDrivingDimension
Dim length14 As Length
Set length14 = constraint28.Dimension
length14.Value = 38.1
sketch5.CloseEdition
part1.InWorkObject = sketch5
part1.Update
Dim pad5 As Pad
Set pad5 = shapeFactory1.AddNewPad(sketch5, 12.7)
Dim limit4 As Limit
Set limit4 = pad5.FirstLimit
Dim length15 As Length
Set length15 = limit4.Dimension
length15.Value = 10.16
part1.Update
Dim reference72 As Reference
Set reference72 = part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());Pad.5_ResultOUT;Z0;G4074)")
Dim sketch6 As Sketch
```

```
Set sketch6 = sketches1.Add(reference72)
Dim arrayOfVariantOfDouble6(8)
arrayOfVariantOfDouble6(0) = 139.7
arrayOfVariantOfDouble6(1) = 0#
arrayOfVariantOfDouble6(2) = 0#
arrayOfVariantOfDouble6(3) = 0#
arrayOfVariantOfDouble6(4) = 1#
arrayOfVariantOfDouble6(5) = 0#
arrayOfVariantOfDouble6(6) = 0#
arrayOfVariantOfDouble6(7) = 0#
arrayOfVariantOfDouble6(8) = 1#
Set sketch6Variant = sketch6
sketch6Variant.SetAbsoluteAxisData arrayOfVariantOfDouble6
part1.InWorkObject = sketch6
Dim factory2D6 As Factory2D
Set factory2D6 = sketch6.OpenEdition()
Dim geometricElements13 As GeometricElements
Set geometricElements13 = sketch6.GeometricElements
Dim axis2D6 As Axis2D
Set axis2D6 = geometricElements13.Item("AbsoluteAxis")
Dim line2D18 As Line2D
Set line2D18 = axis2D6.GetItem("HDirection")
line2D18.ReportName = 1
Dim line2D19 As Line2D
Set line2D19 = axis2D6.GetItem("VDirection")
line2D19.ReportName = 2
Dim point2D18 As Point2D
Set point2D18 = factory2D6.CreatePoint(0#, 63.5)
point2D18.ReportName = 3
Dim circle2D9 As Circle2D
Set circle2D9 = factory2D6.CreateClosedCircle(0#, 63.5, 10.16)
circle2D9.CenterPoint = point2D18
circle2D9.ReportName = 4
Dim constraints6 As Constraints
Set constraints6 = sketch6.Constraints
Dim reference73 As Reference
Set reference73 = part1.CreateReferenceFromObject(point2D18)
Dim reference74 As Reference
Set reference74 = part1.CreateReferenceFromObject(line2D19)
Dim constraint29 As Constraint
```

```
Set constraint29 = constraints6.AddBiEltCst(catCstTypeOn, reference73, reference74)
constraint29.Mode = catCstModeDrivingDimension
Dim reference75 As Reference
Set reference75 = part1.CreateReferenceFromObject(circle2D9)
Dim constraint30 As Constraint
Set constraint30 = constraints6.AddMonoEltCst(catCstTypeRadius, reference75)
constraint30.Mode = catCstModeDrivingDimension
Dim length16 As Length
Set length16 = constraint30.Dimension
length16.Value = 10.16
Dim reference76 As Reference
Set reference76 = part1.CreateReferenceFromObject(point2D18)
Dim point2D19 As Point2D
Set point2D19 = axis2D6.GetItem("Origin")
Dim reference77 As Reference
Set reference77 = part1.CreateReferenceFromObject(point2D19)
Dim constraint31 As Constraint
Set constraint31 = constraints6.AddBiEltCst(catCstTypeDistance, reference76, reference77)
constraint31.Mode = catCstModeDrivingDimension
Dim length17 As Length
Set length17 = constraint31.Dimension
length17.Value = 63.5
sketch6.CloseEdition
part1.InWorkObject = sketch6
part1.Update
Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch6, 10.16)
Dim limit5 As Limit
Set limit5 = pocket1.FirstLimit
limit5.LimitMode = catUpToLastLimit
part1.UpdateObject pocket1
part1.Update
Dim reference78 As Reference
Set reference78 = part1.CreateReferenceFromName("")
Dim reference79 As Reference
Set reference79 = part1.CreateReferenceFromName("")
Dim circPattern2 As CircPattern
Set circPattern2 = shapeFactory1.AddNewCircPattern(pocket1, 1, 2, 20#, 45#, 1, 1, reference78,
reference79, True, 0#, True)
circPattern2.CircularPatternParameters = catInstancesandAngularSpacing
```

```
    Dim angularRepartition3 As AngularRepartition
    Set angularRepartition3 = circPattern2.AngularRepartition
    Dim intParam2 As IntParam
    Set intParam2 = angularRepartition3.InstancesCount
    Dim angularRepartition4 As AngularRepartition
    Set angularRepartition4 = circPattern2.AngularRepartition
    Dim angle3 As Angle
    Set angle3 = angularRepartition4.AngularSpacing
    angle3.Value = 360 / TextBox4
    intParam2.Value = TextBox4
    Dim reference80 As Reference
    Set reference80 = part1.CreateReferenceFromBRepName("FSur:(Face:(Brp:(Pad.2;2)
;None:();Cfl1:());WithTemporaryBody;WithoutBuildError;WithInitialFeatureSupport;MFBRepVersion_CX
R15)", pad2)
    circPattern2.SetRotationAxis reference80
    part1.Update
    End Sub
    Private Sub CommandButton2_Click()
    Unload UserForm3
    UserForm1.Show
    End Sub
```