Faculty of Engineering, Environment and Computing

Department of Mechanical, Aerospace & Automotive Engineering

325MAE - Developing CAD of cooling channels inside a 3D printed mould for a quality aluminium cast

Group 24

Anshdeep Batta - 7548134

Dimosthenis Stratis - 8145659

Tsvetomir Pashov - 7120158

Yash Mehta - 6999791

Contents

Nomenclature

DSS - Dassault Systèmes Solution

COE – Community of experts of DSS

CAD – Computer-aided design

CATIA - computer-aided three-dimensional interactive application

CATScript – CATIA script

VBScript – Visual Basic Script

VBA – Visual Basic for Application

VB6 - Visual Basic 6 Read

CATVBS - Microsoft’s Visual Basic Script

CATVBA – CATIA Visual Basics for Applications

FAST - Function Analysis System Technique

Abstract

One of the benefits of additive manufacturing techniques is the ability to manufacture complex shapes, something which could not be done by conventional manufacturing methods. The companies of aluminium die-casting have seen this as an excellent opportunity for the fabrication of dies with complex cooling channels. Coventry University is currently collaborating with cast Alum company, and the aim is to automate the cooling channel creation process in CAD by using macros. The objectives are to examine the current automation process methodologies and then to develop a macro adapted in a specific function. For this project, CATIA is chosen as the CAD software, and thus all the developed macros are going to be compatible with CATIA. Different scripting languages have been exploited and tested for the creation of the final solution. The solution should be able to generate the conformal cooling channels of the cast by providing the x, y and z coordinates and the diameter of the channel. By automating the process of model designing, it is saving time as the research team needs to repeat the same designing process multiple times during the optimization stage. Furthermore, a less experienced user can work with the macro to create the channels without the need for expertise in CAD. Concluding from the above the novelty of this project is the ability to create cooling channels of an aluminium cast multiple times in less time, with much easier steps and without the need for having an expert background knowledge to execute it.

1.0 Introduction:

In recent years, additive manufacturing techniques are finding more and more applications in the industry; one of them is the industry of aluminium die casting such as CastAlum and Renishaw. One of the main advantages of 3D printing is the ability to manufacture complex components that could have been impossible some years before which in this case are the conformal cooling channels and not only that but can manufacture it in a shorter time than before.

The conformal cooling channels follow the shape of the interior surface of mould at a certain distance through in contrast the conventional cooling consists from straight-line channels made by drilling and not be able to follow the inside geometry because of the limited ability to create different shapes. The conformal cooling solves many of the issues which are resulted due to conformal cooling limitations. One example is the project examined in this paper of aluminium die-casting of automotive components, where the quality of the element is being optimized by using conformal cooling as it is managed to have better control over the temperature. Additionally, it is more efficient by heat transferring means as it can improve the heat transfer rates as it covers more surface compared to a conventional method.

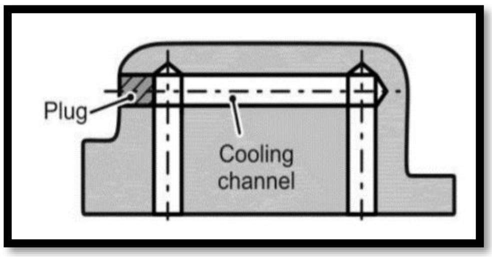


Figure 1: Conventional Cooling made by drilling manufacturing (Ramona Hölker, Matthias Haase, Nooman Ben Khalifa, A. Erman Tekkaya, 2015)

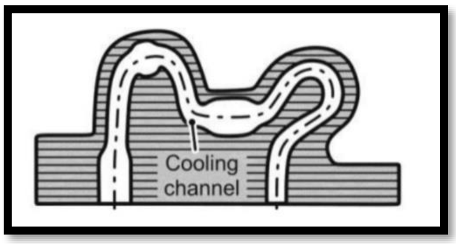


Figure 2: Conformal Cooling made by additive manufacturing (Ramona Hölker, Matthias Haase, Nooman Ben Khalifa, A. Erman Tekkaya, 2015)

This project aims to quickly and easily generate these conformal cooling channels by using a script compatible with CATIA. At the current stage, there is one modelled generated in CATIA and based on this model, and the research team is running heat transfer simulations. According to the outcome, the model is being modified to be optimized. It is a continues process where simulations have been running again and again until the product requirements are met. The procedure of creating or even modifying the model is taking the time, and in some cases, there is need to be done by someone who is not familiar using CATIA. For this purpose, different automation tools such as CATScript, VBScript, VBA, VB6 and CAA have been examined for the creation of an application in CATIA. One objective is to identify what is currently being used in industry and how it can be adapted to this project and in case there are not any similar solutions to how a newly proposed solution can be created.

An existed casted CAD model will be provided in .step file format, and based on that model, the script will be made. Another objective is to use engineering principles to determine the deferent limitations, for example, using solid mechanics to calculate the critical thickness between a cooling channel and the surface which will result in component failure due to cracking formation resulted from high pressure.

An additional objective is the development of multiple alternative solutions and wherein a later stage will be presented to the client, and a decision will be made. Leading to a final selection of which solution will be chosen as the most suitable one and be optimized and discussed again with the client to meet customer requirements.

The layout of this dissertation is as follow:

Chapter 1 isthe introduction of the project and describes the specific problem description, scope, themes, and the novelty involved. In section 2 of Chapter one the Aims and Objectives are demonstrated and chapter 1 is closing with the layout of this dissertation.

Chapter 2 provides the reader with a background of the die casting methodologies and the cooling processes for die casting as well as the differences between conformal and conventional cooling. Closing it explains what the CAD automation process and their applications is. It is It describes an updated and complete critical description of the current state of research undertaken by others in the field and/or closely related fields. Furthermore, it describes and comparing with the approaches undertaken by others in the field, summarizing the gaps and how do they relate with this study aims and objectives and how it is intended to address them.

Chapter 3 presents the research, design and process methods where based on chapter 2 it describes the methodologies employed in this project. In section 2 of this chapter the technical aspects of the solutions are demonstrated as well as the developed solution.

Chapter 4 is analysing the results found from chapter 3 and present them in the form of tables. Later the results are validated with theory from literature review and by the end of the chapter there is a discussion of further improvements.

Chapter 5 is the last chapter of the dissertation and it presents the conclusions by referencing them with the aims and objectives and discuss how the proposed solutions could be improved. Chapter 5 is closing with future recommendations which brings to the end the dissertation.

Next section provides background information about this specific field of engineering and the relevant literature sources that have been identified.

2.0 Literature Review:

The aluminium alloys are finding more and more applications in recent years in industries such as automotive and aerospace because of the benefits which can provide as a material. They are light, and thus a more delicate structure can save energy during manufacturing, corrosion-resistant, which means less maintenance and has a high strength to weight ration which can meet requirements of more demanding applications. The main methods of aluminium manufacture are rolling, extrusion, casting, drawn and forged. This report will focus on casting method and more specifically in High-pressure die casting or HPDC.

In HPDC method the aluminium alloy is in the form of moulted metal, and it is being ejected under high pressure up to 207MPa into a securely locked cavity where it is held until the aluminium has been solidified. As soon as the component becomes fully set, the cavity mould is unlocked, and the movable die-cast opens. The casting component is ejected with that completing one cycle of the process, as soon as the part is ejected. The movable cavity is closed, and the same process is repeated. Depending on the component size and the cooling efficiency, a cycle could last from some milliseconds to some seconds. This has been supported by J.M. Boileau (2010) study where a 110 lb. HPDC V6 engine block has a process cycle time of approximately 90 seconds in 2010, whereas now the cycle is even faster because of the technological advancements. Due to the fact of the improved cycle time, HPDC is able of producing components of different shapes and sizes in high rates making it by that a favorable choice for manufacturing industries.

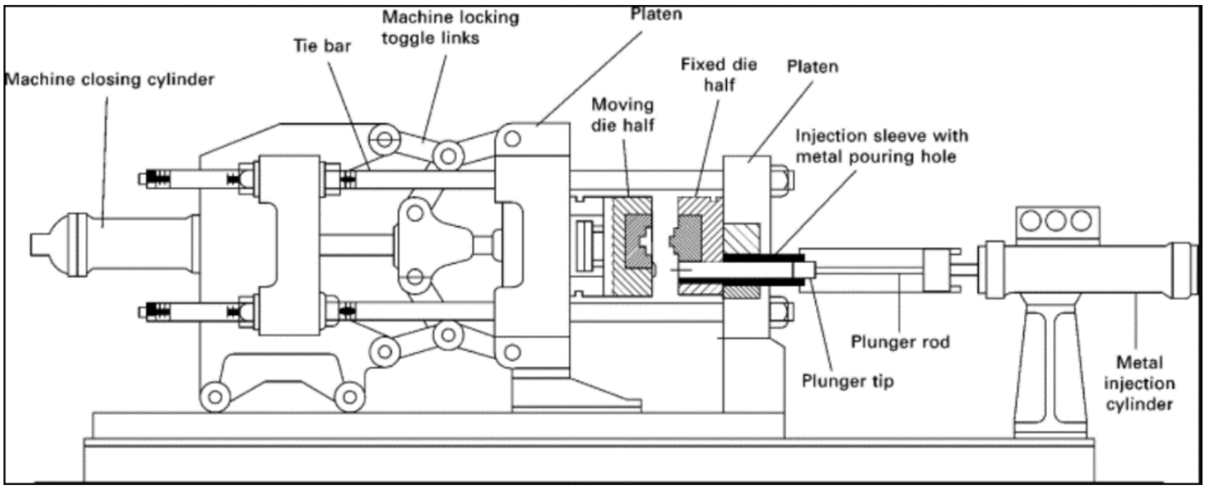


Figure 3: High-Pressure die casting machine Materials, Design and Manufacturing for Lightweight Vehicles

Another significant advantage of the high pressure die casting is the excellent surface finish results due to the high pressure which enables good contact of the moulted alloy and the die surface, achieving by that better cooling rates. The benefits of a good surface finish are the fact that no further surface treatments are necessary therefore saving time by reducing man-hour and reducing the final cost of the product because there is no need of extra processing equipment. The HPDC method has excellent dimensional accuracy because of the tight tolerances resulted from closed cavities which means that there are minimal defect components. Also, the parts are characterized by improved hardness and strength due to the high pressure applied. Closing HPDC is able of producing highly complex components with thin walls achieving the same strength with less weight.

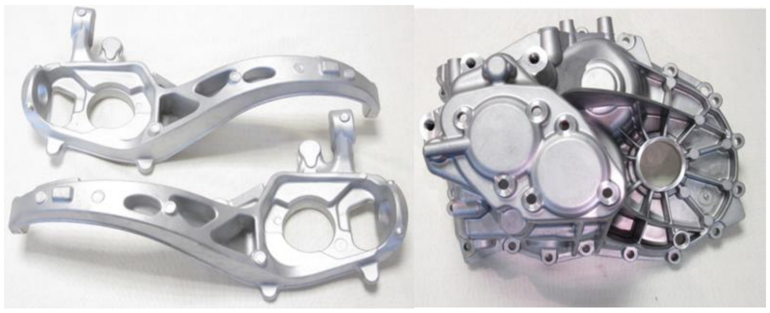


Figure 4: Aluminum alloy components for automotive applications made by CastAlum company

Despite all the advantages, some disadvantages are resulting from high pressure. The equipment is complex, die cavities are very costly to be manufactured and thus is requiring a large capital investment. Cycle time cannot be changed according to the demand which means if the order is low, it is not possible to produce at low rates and by that making it not suitable method for low cycle production, so it makes it not profitable to start with small quantity production. Another challenge to be faced is the formation of vortices during the injection could form trapped air in the component resulting in stress concentrations. Still, the recent years' different solutions have been introduced to eliminate this phenomenon, for example, there are vacuum systems where they are able of reducing the amount of entrapped air, another method introduced is the fill-in slower rates to minimize the turbulence which is known to form entrapped gas. Additionally, during the solidification, the component could suffer from shrink holes which are formed due to shrinkage during the cooling stage.

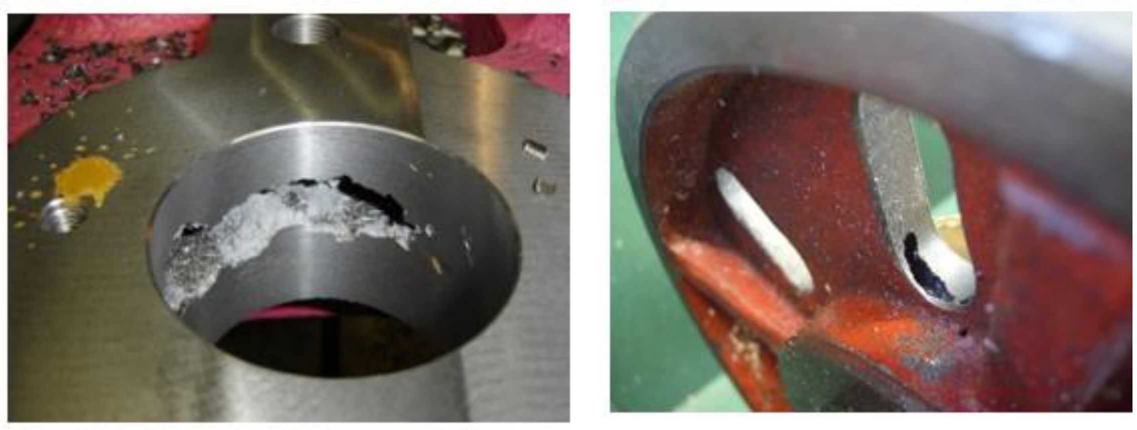


Figure 5: Shrinkage defects during solidification

The cooling stage during the cycle is greatly affecting the quality of the component as well as the cycle time, the better the cooling stage efficiency is, the better the results are. Thus, conformal cooling is a crucial player for aluminium dies casting manufacturing. Studies have been carried out by Ahn DG (2011) to compare the results between conventional and conformal cooling for parts with complex geometries for which the traditional straight-line cooling channels are not able to be placed closed to mould surface. In contrast, conformal cooling channels can follow the mould surface path lines retaining a constant thickness throughout the surface geometry. Another case study carried by moldex3D has been examining the two cooling techniques, and it has been found that the cooling distribution in the conformal case was much more uniformly throughout the component with a temperature difference of 2-3°C.

In contrast, the temperature difference in conventional cooling was around 5-7°C. Furthermore, it has been found out that the max surface temperature of the traditional cooling was around to 84-91 °C whereas in the conformal cooling was around to 80.5-81.5°C which this means more cooling efficiency as it manages higher heat transfer rates in conformal cooling. The discussed findings can be seen in the below figures.

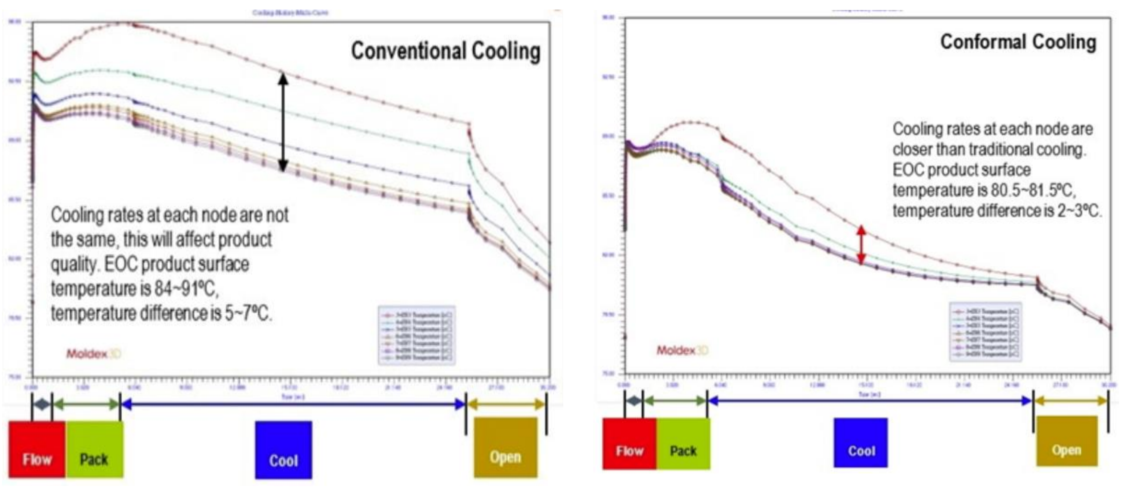


Figure 6: Temperature distribution of Conventional VS Conformal cooling. (Moldex3D)

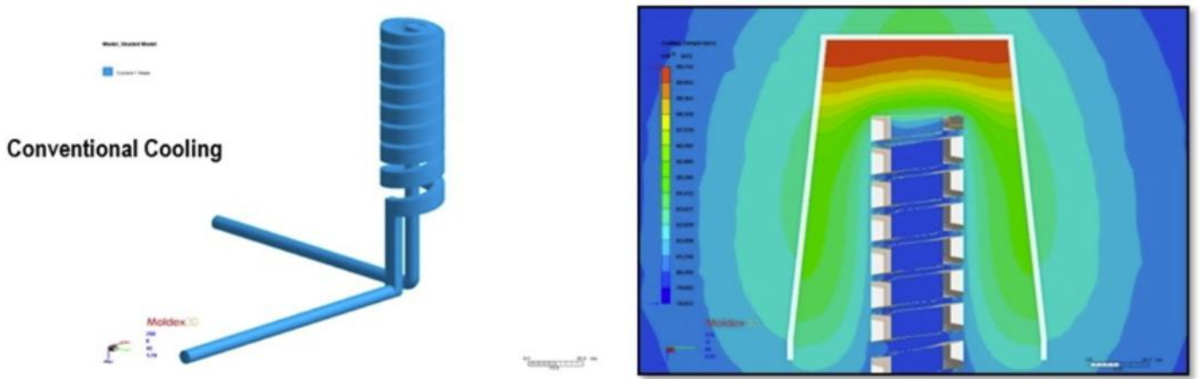


Figure 8: Temperature distribution for conventional cooling (Moldex3D)

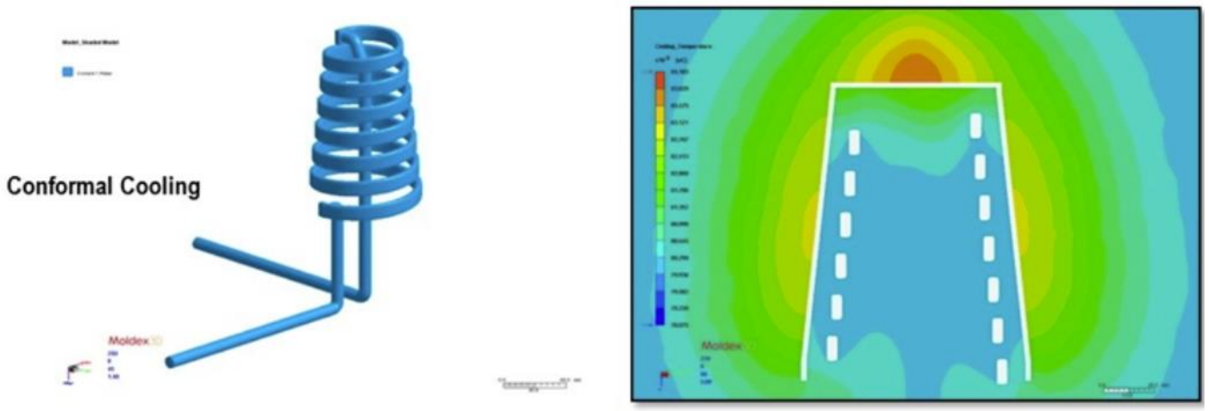


Figure 7: Temperature distribution for conformal cooling (Moldex3D)

The cooling phase plays a critical role for component quality and cycle time as it is shown in the below figure for a typical cycle time the cooling step is almost the 50% of the cycle time an the lower this time is the fastest the cycle time is. Thus, the bigger the production capacity will be. Still, at the same time, the cast has to be fully solidified before die clamps unlocked, and the way to achieve this is by introducing the optimized conformal cooling channels design. From place to place the diameter of the channel vary as there are different cooling demands for each region of the component and that results in a uniform heat transfer. Thus, the part is uniformly cooling, which means that material properties will be the same everywhere in a different case where there is no constant cooling; there is the risk of varying material properties. In some areas, the alloys would be softer than other sites resulting in unpredicted failure. Furthermore, by distributing the temperature uniformly, there is better control of the material shrinkage, which results in shrinkage holes during the alloy solidification resulting in fewer defect components, thus better dimensional quality.

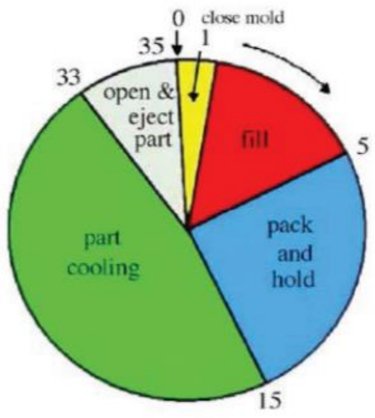


Figure 9: Typical cycle time (Suchana A. Jahan, Tong Wu, Yi Zhang, Jing Zhang, Andres Tovar, Hazim Elmounayri, 2017)

Extended Literature Review on Automation Process Using CATIA:

The goal of this literature review is to understand the problems faced, possible outcomes, and solution methods available to successfully create a macro in an engineering design software called – CATIA. This, in return, would help to develop a CATIA script and graphical user interface for a user who can then create a cooling channel within an intricate geometrical part.

Ziethen (2013) provides a brief introduction on the CATIA v5: macro programming discussing CATScript, CATVBS and CATVBA. Although the book only deals with MS Visual Basic Script, it does provide a few examples for Visual Basics for Applications. The author is a known personality in the field of CAD systems engineering, has experience as an IT consultant at MAN Truck and Bus AG, and teaches the CATIA software at Munich University. The book contributes to suggesting distinctive communications methods via practical examples, numerous case studies, and sample macros. It is a combination of conceptual ideas that illustrate the abilities of the automation process using macros in CATIA. It is mentioned that these programming languages effectively work with methods and objects in CATIA. Objects store the information; they act as a container. Example of items is any CATIA part, line, surface. Whereas methods help to implement instructions which initially creates the objects. This book acts a guideline for the initial and straightforward questions which a beginner will face during the process of creating a macro in CATIA. The biggest limitation of the book is the sample macro codes provided only work in CATIA R19 or higher versions.

Automation in any designing software can be done using macros. This is done to save time, energy, avoid manual errors, increase productivity, reduce labour cost and other related expense. The growth of automation is snowballing. It is readily applied in many engineering fields like aerospace, mining, highway systems, waste management, etc. Siddesh S and B.S. Suresh (2015).

Automation in CATIA is a feasible process and has been carried out by various experts in the field. The reason for automation is different for all users. This depends on the expert knowledge and industries where the solutions to be applied. Modelling in solids is a task where countersunk holes are created on an aerospace part by automating the process using visual basic for applications (6.0). Gloria Del Río-Cidoncha, Martínez-Palacios, and Ortuño-Ortiz (2007) have used several generic inputs to accomplish this task. The CATIA tools like power copy, user feature and macros assist in automating this process. From their experiment, it is also proven that visual basics for applications software allow a user to create a quick and efficient macro script. And macros are one of the most potent tools which CATIA has to offer as an engineering design software. A user form is created to input values and make the centre point for holes in CATIA. The figure 1 below gives a brief idea about how the points are created in CATIA. The values/coordinates for points are entered in the software using the user form generated using visual basics for application (6.0). Hence this automates the process of generating lighting holes. Lighting holes in any structure helps to reduce the weight without causing a significant change in structures dynamic properties. The focal point of this solution method is using multiple interactive windows/message boxes, and generation of planes refer to figure 2.

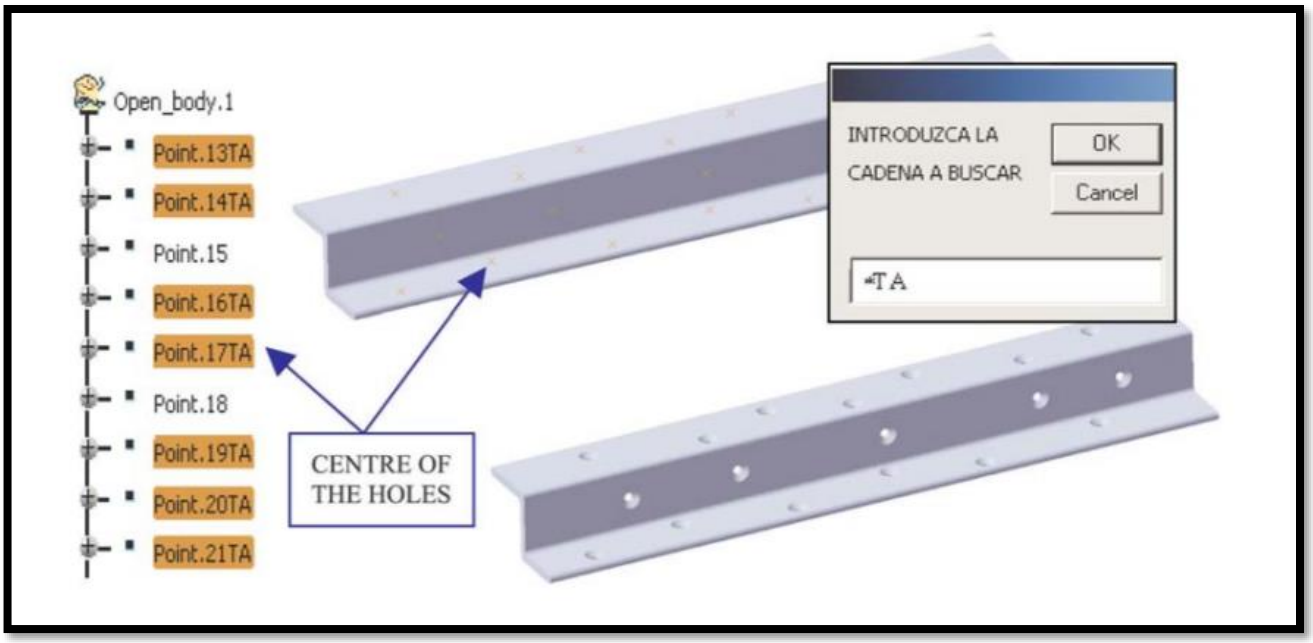


Figure 10: Example of user form and lighting holes creation (Gloria Del Río-Cidoncha, Martínez-Palacios, and Ortuño-Ortiz 2007).

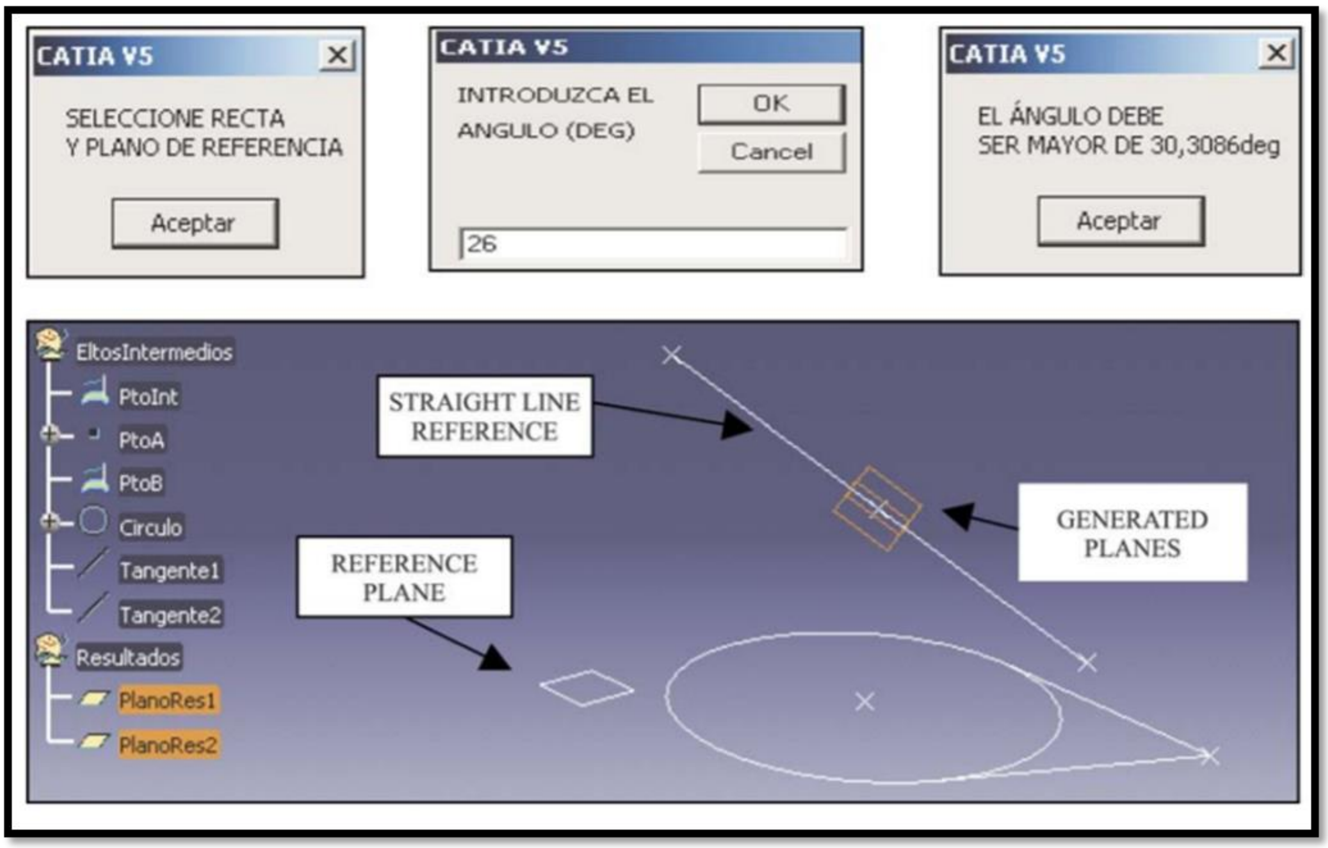


Figure 11: Focal point interactive windows and generation of planes (Gloria Del Río-Cidoncha, Martínez-Palacios, and Ortuño-Ortiz 2007).

A method applied by Kumbhar, Pawar, Jadhav and Dhanrale (2014) gives a brief idea about customization of CATIA for creating different types of holes. In this case, the holes are made on disc wheel. The idea of recording the macros in CATIA software is an essential part of the scripting. The authors have used a series of function in macros where scripts are written in visual basic applications. The use of the CATIA part document automation objects tree provides a base for customization structure of the macro. The CATIA part object document tree has two groups of components. The first is related to the geometry, which includes planes, part bodies, geometrical sets and other attached geometries. Whereas the second one is a group of metadata example part number, standard attributes, customized value attributes and nomenclature.

A perfect example of defining PartDocument in macros and their importance is highlighted by Siddesh S and B.S. Suresh (2015). Figure 3 attached below roughly provides an idea of programmatic structure which should be followed to script a macro. This is done to access an object part property of any CATIA PartDocument. A function such as PARTDOCUMENT.Part AS Part (Read-Only) can be used.

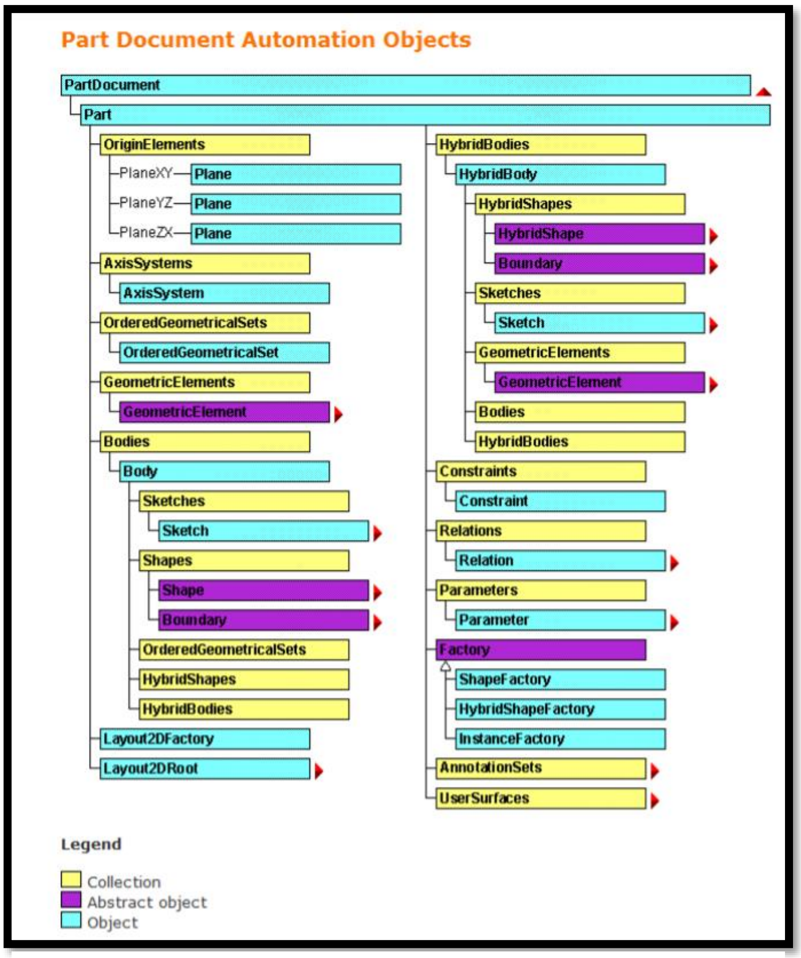


Figure 12: Example of the part document tree (Siddesh S and B.S. Suresh 2015).

Siddesh S and B.S. Suresh (2015) generate a helical spring coil in CATIA refer to the figure below.

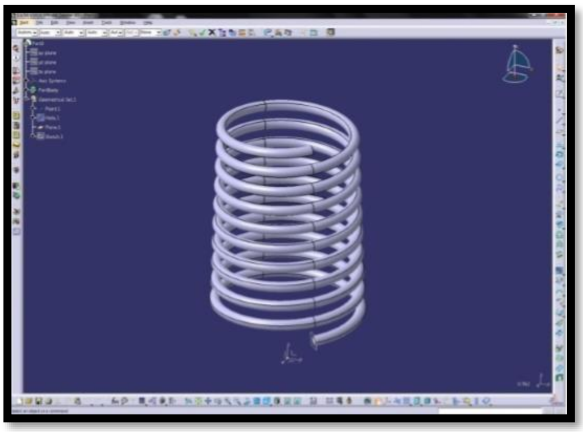
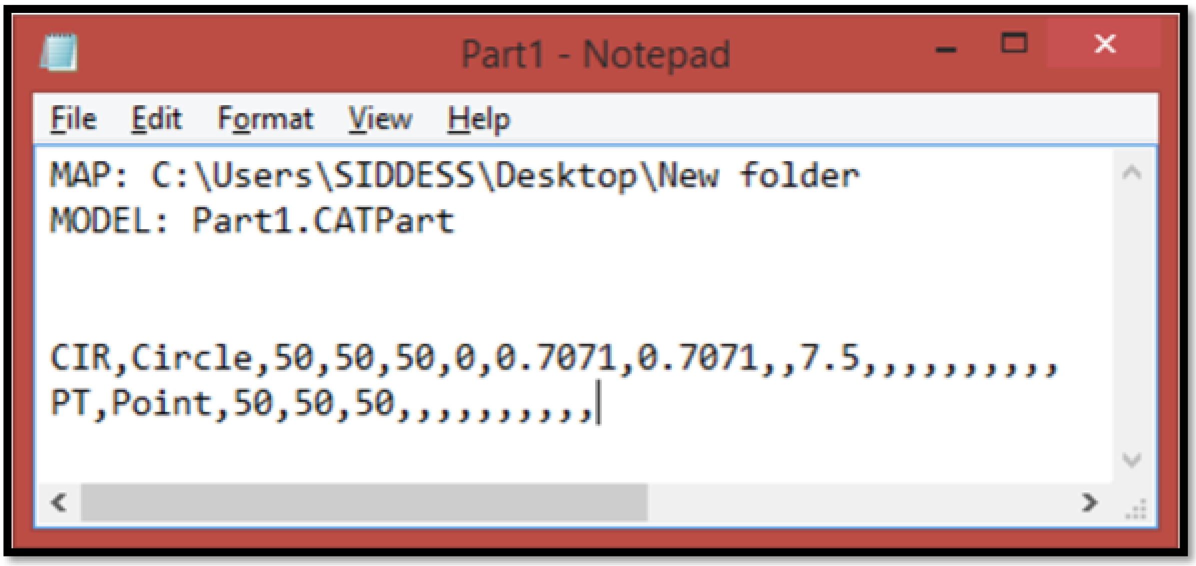


Figure 13: Generation of helical spring coil in CATIA Siddesh S and B.S. Suresh (2015).

Using the recording function, the author records their repetitive task. A user form is then created, which allows them to view or save the file. The file is a CSV format file, and which can be opened using a note pad. The text file gives the permissible feature type of the spring. Hence the editable features can be altered to generate new similar coils. Features like starting point and diameter of the spring are editable.



```
MAP: C:\Users\SIDDESS\Desktop\New folder
MODEL: Part1.CATPart


CIR,Circle,50,50,50,0,0.7071,0.7071,,7.5,,,,,,,,,,
PT,Point,50,50,50,,,,,,,,,,
```

Figure 14: Example of editable text file of spring coil Siddesh S and B.S. Suresh (2015).

A recent example of automation used by Ali, Shafie, Dharmalingam, Daud, Zakaria and Latif (2018) is an exciting procedure which creates a CAD model for rim wheel. The automation method gives focus to the essential parameters and features of wheel design. Figure 5 shows the steps followed by the authors in developing a macro code in CATIA.

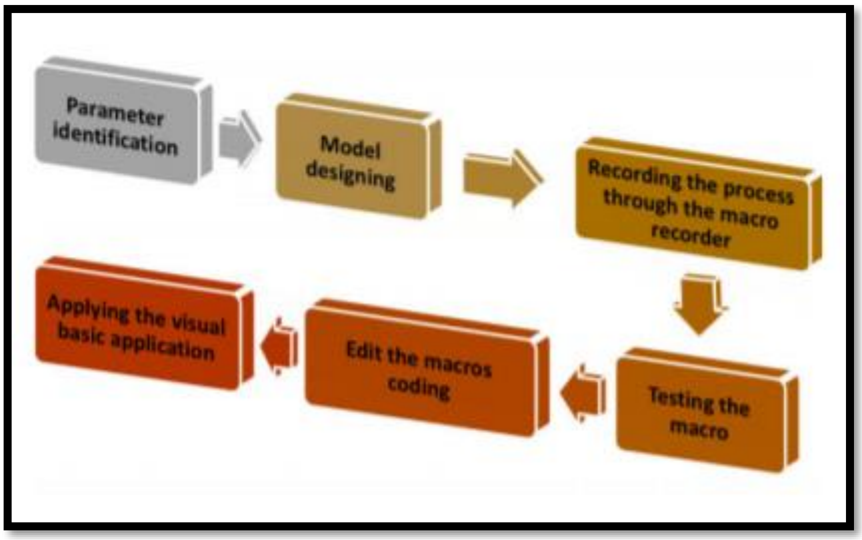

Parameter identification
Model designing
Recording the process through the macro recorder
Testing the macro
Edit the macros coding
Applying the visual basic application

Figure 15: Flow diagram indicating the main process of automation in CATIA (Ali, Shafie, Dharmalingam, Daud, Zakaria and Latif 2018).

This method can also be called as trial and error, which is a known problem-solving method. Here macros are recorded while designing a part/geometry. Macro recording tool is one of the essential tools available in CATIA. The recording script is then tested, and changes to important parameters are made. The initial designing script is used as a reference. A parameter interface/user-defined forms are added to change the dimensions and features of rim wheel directly. A critical point mentioned by the authors is during modelling use of functions such as undo should be performed this done to avoid complications in the algorithm recorded. User-defined forms are created using visual basics, but the procedure to create the forms is not shown. Although, example, images do depict the possibilities of creating a graphical user interface.

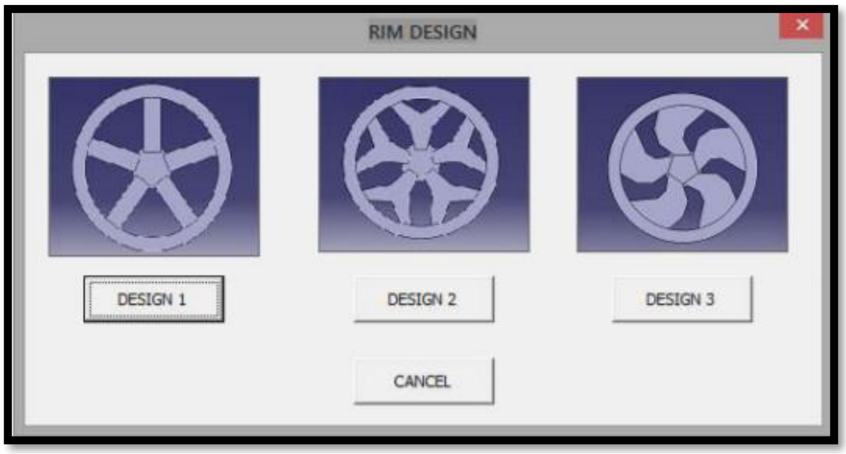
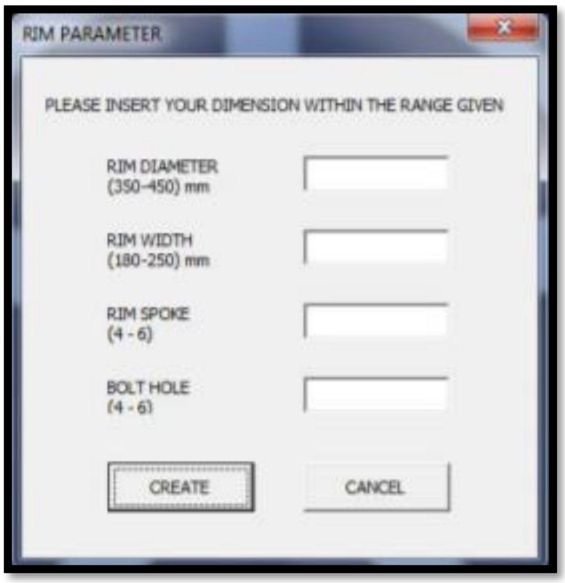


Figure 16: Example of user-defined forms (Ali, Shafie, Dharmalingam, Daud, Zakaria and Latif 2018).


RIM PARAMETER

PLEASE INSERT YOUR DIMENSION WITHIN THE RANGE GIVEN

RIM DIAMETER (350-450) mm

RIM WIDTH (180-250) mm

RIM SPOKE (4 - 6)

BOLT HOLE (4 - 6)

Figure 17: Example of a graphical user interface to input parameters (Ali, Shafie, Dharmalingam, Daud, Zakaria and Latif 2018).

The experiments above performed by various experts in the CAD designing field all have one thing in common is recording the initial designing of the model/geometry and analyzing the algorithm created by the macro recorder. Because there is no exact guide available to script the macros in CATIA, it is indeed a complicated job to script a new concept.

As the documentation and examples for creating macros in CATIA are relatively low. From the above research, it can be said that macros in CATIA are used to reduce the workload of a CAD design engineer. They are more efficient when the initial process of designing a part is recorded, and the overall purpose performs the task repeatedly. Parts should have the same geometrical features, but it is possible to alter the dimensions of the overall piece. The biggest drawback of recording feature is deleted or undo function causes errors in the macro script. It is challenging to design an intricate part without the use of delete or undo function.

Next section shows the approach taken into developing initial approaches to creating a solution and the 2 most suitable solution methods that have been chosen.

3.0 Research / Design / Methods Review:

Solution development

For the solution development for this project, the engineering principles of the design process have been used. The flow diagram below is making sure that the final product is going to meet customer requirements and to avoid the repeating or doing unnecessary steps or even missing any critical steps. The first step is to identify the customer needs, which has been complete on the first meeting. At that stage, the team had a meeting with the client, and the client requirements were documented. The next step was to research the market and current solutions for the same or for similar applications which have been analyzed in chapter 2 of this paper (literature review). In the next component, multiple steps are taking place at the same time, which are specifying the design constrains, criteria and requirements. Following steps are around of solution attempts and concept solutions which are further explained in this chapter. After these steps, the possible solutions to the problem are starting to take shape, and it is clearer how the solution would look like. In chapter 4, the steps of proposed solutions will be developed, and design in detail and by the end will be compared evaluated. A decision will be taken of which will be chosen as a final solution.

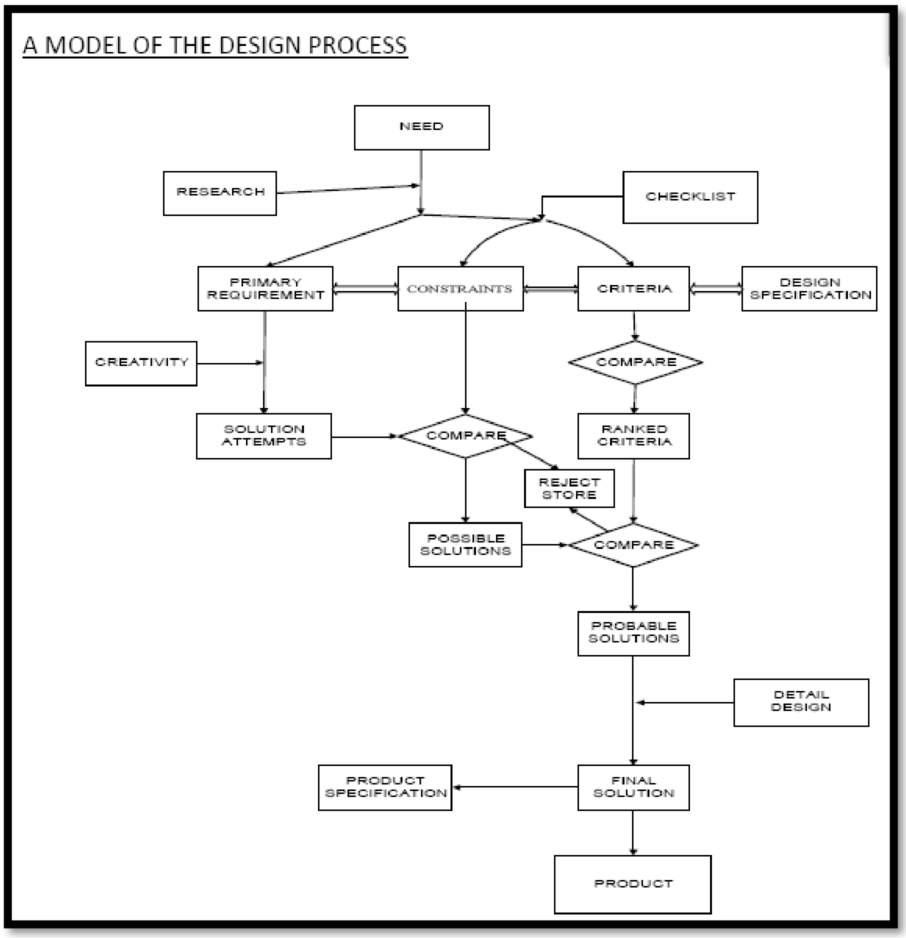


Figure 18: Model of the design process

FAST, available in Appendix A, have been used to identify what are the project objectives and illustrate how the proposed solutions are going to meet the project requirements and client's requirements. The higher-order function for this project is the automation of the extrusion process, which is made in CATIA. To achieve this higher-order requirement, a script must be developed which will be compatible in CATIA. There are two main routes for creating script/macros, either by importing the CAD model into scripting language software or by recording the macro inside the CAD software. For the first case, the script commands must be created from the beginning, whereas in the second case, the design tree is already be done from the CAD software. As soon as the CAD modifications have been made by the user which are the x, y and z coordinates and the diameter the script has to be imported back to the CATIA for the first case. In contrast, in the second case, this step is skipped as the macro is being on worked from inside CATIA. By the end of the process, the client requirements have been met by having as lower-order function the time saved due to the automation process.

Problem Description

Gantt Chart, available in Appendix B, is a useful way of communication within a project as it helps in prioritizing the tasks and projects on urgency. Gantt Chart helps in assigning the tasks to the resources available, tracking the process of the project and making the sure that the functions are not overlapping each other. It also helps in analyzing the critical path of the project to make that the project is completed within the given deadline.

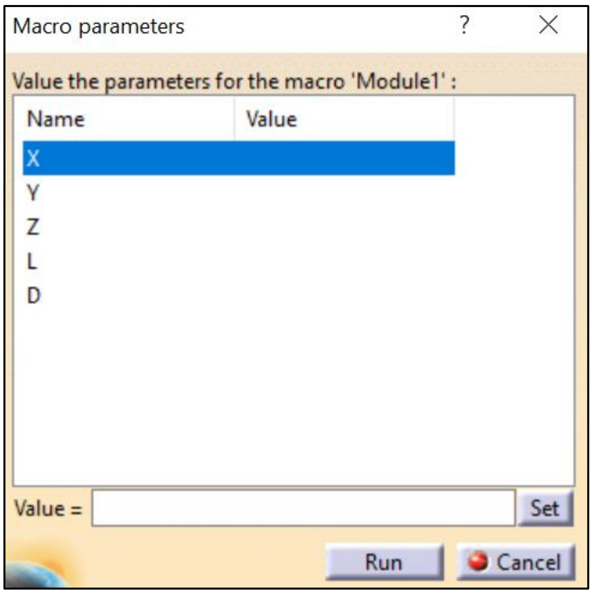
According to the Gantt, all the tasks have been completed within the expected time frame. Due to global pandemic COVID-19, the deadline was extended by three weeks. Therefore, the Gantt chart has been updated according to it. While working on the project, the milestones depicted the progress of the project, which are marked with a red flag. As from the Gantt chart, the first critical milestone of the project was to acquire information on the CAD model from the client, which gave a better understanding of the project and creating an initial client presentation. The next milestone was about researching the current methods used in the industry to cool the die, as mentioned in the literature review. Understanding the concept of the modern techniques helped in doing CAD automation to extrude the channels for the analysis. To begin with the CAD automation, it was essential for the team to have a basic understanding of macros and visual basic and how it can be linked with CATIA to have an automated extrusion process. As a result, Learning Visual Basic was considered as the third milestone. The fourth milestone of the project when the project was in the middle stage was implementing the macro developed on a model to make sure the macro code runs without showing up any error. After fixing up all the errors in the macro and making sure that the file can be exported to Ansys without any mistake. The fifth milestone was the optimization of the final solution to make it more user-friendly to input the data and to access the macro code as discussed in the initial client meeting. All these milestones needed to be met before the deadlines as extending the milestones could delay the finish date of the project. The team did face some difficulties while completing their task due to lack of expertise and the global pandemic as there was no access to the books in the library due to lockdown. Still, we supported each other to complete the task and finish the project successfully before the deadline.

Methodology

The software used to create the automation of extraction of channels is Dassault Systems, a French Company have developed CATIA V5 (Computer-Aided Three-Dimensional Interactive Application). CATIA was developed in the C++ programming language and was the begging of the company. CATIA V5 is a complete engineering toolset within a single working environment. It is mainly used to design, simulate, analyse and manufacture a variety of products in different industries such as aerospace, automotive, consumer goods and industrial machinery. It is used from all manufacturing companies, from OEMs through its supply chains, to small independent producers. CATIA is being chosen as the primary 3D design systems.

First Solution

This was a brainstorming idea of creating a channel through one of the faces of the cuboid. Cuboid was considered as a prototype of one of the aluminium casting. In figure 1, five different parameters can be set by the user. X, Y, Z, L and D where X is the coordinate for the x-axis, Y is the coordinate for the y-axis, Z is the coordinate for the z-axis, L is the length of the extruded channel and D is the diameter of the channel that is required to be extruded.


Macro parameters

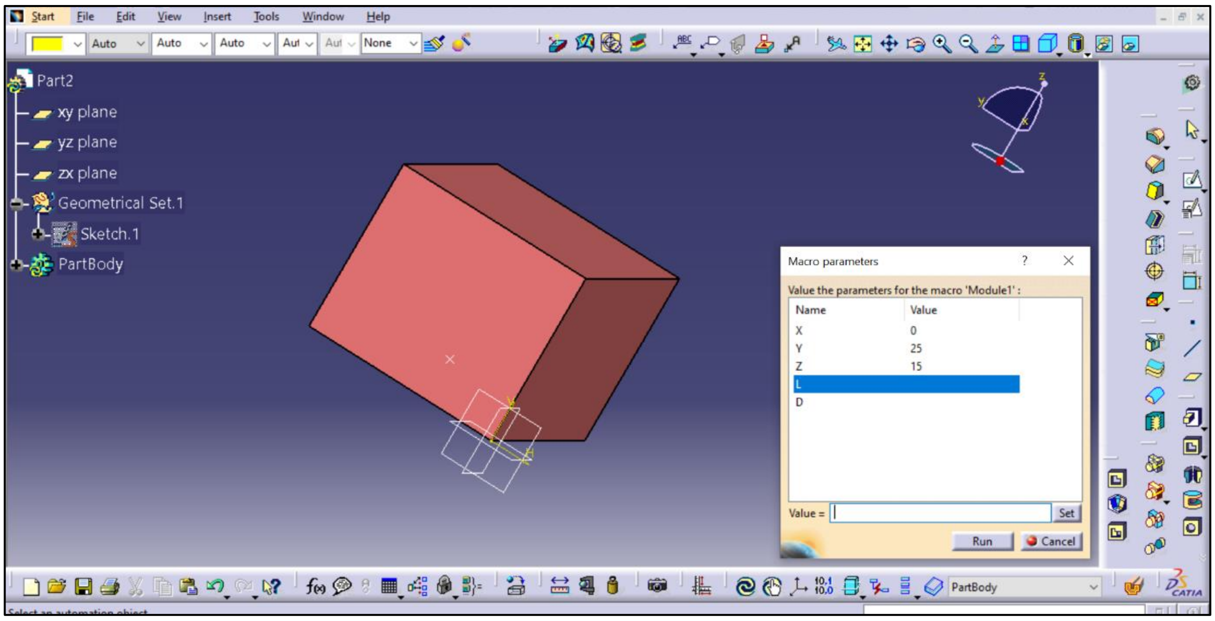
Value the parameters for the macro 'Module1' :

Name	Value
X	
Y	
Z	
L	
D	

Value =

Figure 19: Parameters for the extrusion

In Figure 2, X, Y and Z are have been given by the user, and a point has been created in a 3D global coordinate system. Considering this point as the centre of the circle as a circle will be generated after inputting the D, which is the diameter of the circle. The X, Y and Z point are calculated from the origin of the body. This parameter has been created using the command: Sub CATMain (X As Integer, Y As Integer, Z As Integer,



Macro parameters

Value the parameters for the macro 'Module1' :

Name	Value
X	0
Y	25
Z	15
L	
D	

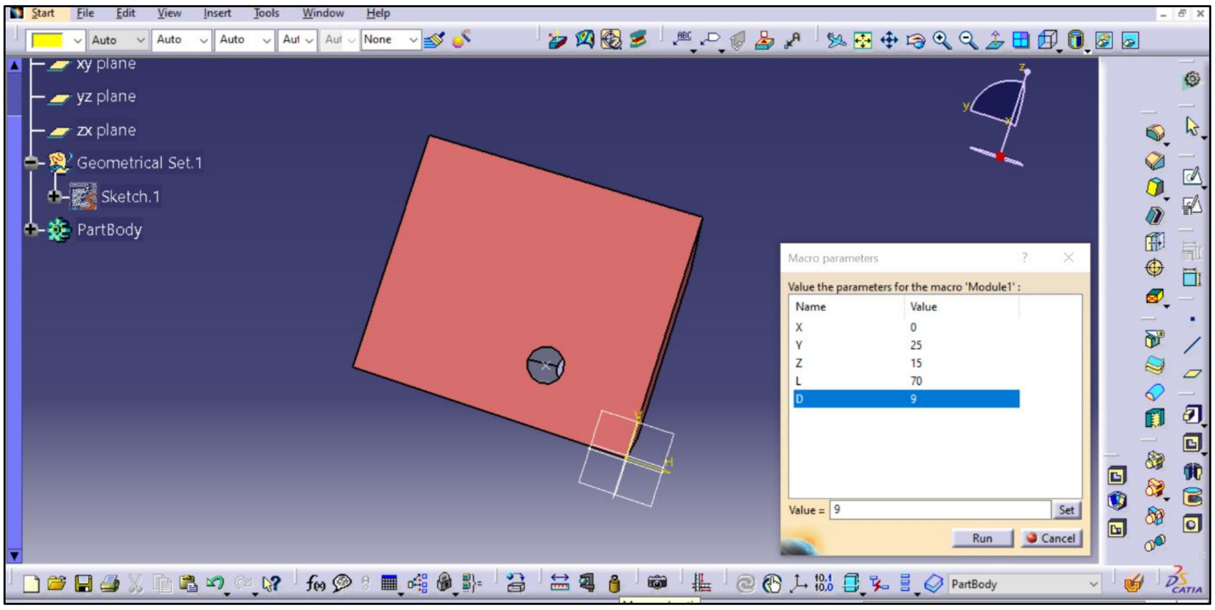
Value =

Set Run Cancel

Figure 20: Inserting a point in 3D Global Coordinate

L As Integer, D As Integer) which helps in creating the above parameter box and the values input by the user can only be integers, or else the code will not run.

In figure 3, the completed extruded model has been generated from all the parameters input from the user.



Macro parameters

Value the parameters for the macro 'Module1' :

Name	Value
X	0
Y	25
Z	15
L	70
D	9

Value = 9

Set Run Cancel

Figure 21: Completed Extrusion Process

This macro was created using recording tool in the CATIA. After recording the macro, there were some changes made in the recorded to put the parameter box and make customized extrusion channel. Some of the macro codes were changed to match the values input from the user with the code. The limitation of this method was only one extrusion can be created at a time, and for multiple extrusion, the user needs to repeat the step every time. Also, another limitation is that to start the extrusion, the point considered should always be on one of the dies. If the point considered is inside the die, then, the macro won't run and end up having an error and stopping the code in the middle. As the code can't generate a plane inside the die, therefore the point can be considered inside the cube. As a result of which either X or Y coordinate needs to be regarded as zero so that the point is on the face.

Second Solution

The internal layout of the cooling system of the die-cast is based on generating individual channels and connecting them one at a time.

The concept uses datasets present in a spreadsheet. The datasets parameters are forming the inner layout of the cooling system (channels). An individual dataset is just a single row inside the sheet and contains 4 parameters – x, y, z and radius. The x, y and z are used for reference start and endpoint of a single channel and the radius as its z dimensional property.

Creating the channels based on the given sets can be complicated if the initial approach doesn't break the tasks properly. Because of that, a simple starting method of using a for loop that extracts 2 consecutive datasets and performs single-channel generation on them was identified. The single channels are created with a connection at both ends ensures that the layout will always be fully connected.

A schematic that represents the approach can be seen in Figure 17.

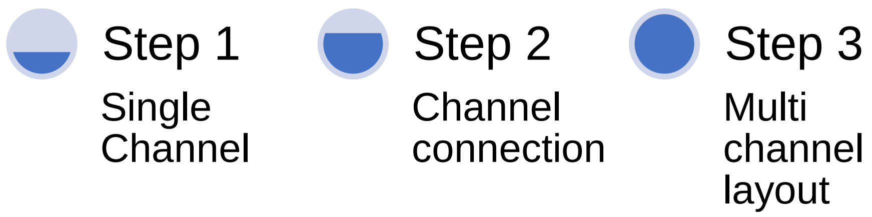


Figure 22: Basic structure approach

This method focuses on creating a single channel as the initial step and then creating a connection at both ends of the newly created channel. When the second, third, etc. channels are generated, they will automatically form the inner layout of the cooling system. Preventing any additional work to be done for improving the cooling structure. Making the overall system highly accurate regardless of how far down the sheet the method has been executed. A slight problem based on this approach is related to the connection. The connection represents a spherical cut inside the solid geometry placed where one channel starts and other ends. The spherical link was selected because a round connection will have lower stress concentration at a single point. In contrast, a sharp connection will have higher stress at a single point, usually the sharp edges. This can lead to unwanted and excessive stress creating crack inside the cooling system.

Method

The pseudocode of the technique can be seen in Appendix C and the explanation of the below.

This solution focuses on generating a channel that is entirely inside a solid geometry, unlike the first solution that is based on making a pocket starting from a chosen surface. Because the fully embedded channel needs to be designed in such a way that it doesn't require any contact with a surface as a reference but make its own reference. Leading to the development of the approach described below.

Step 1 - Single channel generation

The sub-method works with the following input:

- X1, Y1, Z1 – as a start point of the channel
- Radius 1 – the radius at the start of the channel
- X2, Y2, Z2 – as an endpoint of the channel
- Radius 2 – the radius at the end of the channel

The way the approach works is by making a plane in 3D space that will be used as a reference of a channel end regardless of its distance from any surface leading to creating a plane with origin point at the given x, y, z components for both the start and end of the single channel.

Using the x, y and z values of the datasets a plane will be created that is offset to the other end of the channel.

It is making the positioning of the plane exactly at the x, y, z values and having its origin point (0, 0, 0) to be referenced those values.

On the generated plane, a circle with the specified radius will be created.

The plane and circle generations are done at both the start and the end of the channel.

It is then followed by creating a pocket-like structure that will remove the solid material between the two circles at the start and end of the channel.

This summarizes the generation of a single channel.

A possible addition to the solution can be the difference in diameter of the channel from the start to the end, such as narrowing or enlarging one of the ends. Such addition will require additional research.

At the end of the single-channel generation, connection sections will be manufactured, as stated in step 2.

Step 2 – Connection

The connection of the individual channels is based on using the point where two channels meet. This is usually (always) where one channel ends and the next one starts. This is done to improve the structural integrity of the die and reducing the stress concentration present at sharp edges. The spherical solid removal is created using the radius value of the adjacent ends of the channels and performing a solid cut. This leads to removing the round section from the solid and making the final channel connection.

Like in the previous step this is completed for both ends of the channel.

Step 3 – Multi-channel layout

Uses a for loop to go through the sheet and calls the single-channel generator with rows(ii-1) as a start point and line (ii) as end row where ii is an array of integers from 2 to the length of the sheet.

Next section provides a detailed description of the developed solutions, how they work and what are their dependencies. It also points out the difference between them and based on what a final solution has been chosen along with challenges along the way of the development curve.

4.0 Results, Analysis, Discussion

Solution 1:

The code for solution 1 is given in Appendix D, and a brief explanation of the code is given below.

The code is executed with Sub CAT Main which is the necessary command line to run any macro code.

It is generating the Parameter as User Interface between the code and the user to make the extrusion process quick and straightforward.

The code was:

Sub CATMain (X as Integer, Y as Integer, Z as Integer, L as Integer, D as Integer)

Where X can only be an integer value and signifies the x-coordinate, Y is also an integer value indicating the Y-coordinate, Z as an integer value referring the z-coordinate, L as Integer defining the length of the extrusion in millimetre and D as the diameter of the extrusion channel.

The code behind the automation is simple in terms of understanding as it takes the input from the user, and based on the x, y and z coordinates, it creates a point. The x, y and z coordinates on the object depend on the geometry of the model as from where the origin (0,0,0) of the model is located. In some models, it can be in the middle of the geometry if the model has been created considering the origin point in the middle of it can be made from any corner side. This must be considered before inputting the point coordinates. Also, the other important thing that needs consideration while inputting the coordinates is that either X or Y coordinates need to be zero as the extrusion can only take place if the point is generated on the surface of the model which is one of the limitations of this solution which is discussed further in the Discussion And Conclusion section of this report. After the point has been created, it automatically generates a circle with the sketch option and required diameter in millimetre in CATIA and coincides the centre of the circle with the point. Using the pocket function in CATIA toolbar, the channel is extruded based on the value input in the parameter box.

Solution 2:

The pseudocode of the method can be seen in Appendix C and the explanation of the below.

As shown in Appendix C and the description of the initial approach in methodology, many of the sections and subsections are the same. The differences are those that support the execution of the significant components of the macro. Like stated before the solution is based on generating channels that are entirely inside a solid geometry removing the need for any contact with the surface as a reference. Below is the detailed explanation of the method.

Step 1 - Single channel generation

The inputs remain the same:

- X1, Y1, Z1 – as a start point of the channel
- Radius 1 – the radius at the start of the channel
- X2, Y2, Z2 – as an endpoint of the channel
- Radius 2 – the radius at the end of the channel

The method works by creating a plane is in 3D space at the specified coordinates by using them as a reference. This plane is made for both ends of the channel and is created regardless of its distance from any surface.

The plane is created using the following sub-method

Using the x, y and z values of the datasets, a reference point in 3D place is created. Additionally, the values for a plane equation are calculated based on A*x + B*y + C*z = D where A, B and C represent the 3D values of the vector between the generated start and endpoint in 3D space of the single channel. (DataSet2 – Dataset1 for x, y and z).

The plane is then created using the equation value to support the tilt of the plane in all 3 axes. Because there were some issues with the positioning of the plane when using only the equation, the reference point on 3D was required to be used as a Reference. This has led to setting the plane origin point (0, 0, 0) to be referenced exactly at the x, y, z values to be.

The constructed plane is used as a reference on which a circle with the specified radius in the dataset is created.

The procedure of produced planes and reference circles is completed at both the start and the end of the channel. Followed by the "Remove multi solid section" operator using the two circles as references and removes the solid between with unselecting the twisting option from the menu. This concludes the extrusion of solid cut inside a 3D geometry essentially mimicking a channel.

Using the "Multi solid section" the difference in the diameter at the ends of the channel can be neglected because it is incorporated and will not trigger any errors or miscalibration when creating the channel.

This step is followed by making the channel connection in step 2.

Step 2 – Connection

Connecting two adjacent channels is vital because the cooling fluid needs to pass through them and cool the die as intended. For that purpose and as previously described a spherical connection is used. The location of that connection is at the meeting point of 2 or more channels, meaning it is the point where one channel ends and another starts. For a link of more than two channels, the connection is the same with the exception that multiple channels pass through it.

The link essentially represents a spherical solid cut or material removal with the radius value of the adjacent end of the channel executed using a Groove cut. The Groove cut uses a sketch of a half a circle and a diameter line of the circle as an axis of rotation. This leads to removing the spherical section from the solid and creating the final channel connection. The cut method is carried out at both ends.

And as mentioned in the technique overall description if more than two channels use the same link the connection itself will have a radius equal to the highest z dimensional property of all the channel passing through it. This will only have an implication of the channel diameters are not consistent through the layout.

Step 3 – Multi-channel layout

The same concept as the one described in the methodology section has been kept. Hence no changes to it have been made.

Assembly of methods

The assembly of the methods used in solution 2 can be divided into three sections.

- Identify the part in which channels will be generated
- Identify sheet containing the locations of the channels
- Perform the channel generation using the three steps

Identification of part that will be edited is the first component of the assembly. It focuses on identifying if there is a part that is currently active in the workplace, and if not, it prompts the user to select one from the pop-up file selection window. This is created in such a way because the user could be editing or designing a geometry that hasn't yet be saved or is currently in progress.

The second component is again another file selection. This file selection is the datasheet file and is compatible with .csv or .xl* file extensions. If no file has been selected, the component will create a new datasheet and inform the user to populate it. The spreadsheet was selected as the ideal method for inputting dataset on a large scale.

Finally, the last component of the assembly is a for loop that executes the channel generation and channel connection described above with two datasets as a starting point.

Challenges

Marco challenges:

Throughout the project, numerous challenges were met in terms of working with macros. Those challenges can be summarized in the following areas:

- Low accessibility to documentation

- Complicated documentation
- Incompatibility between methods and variables
- Insufficient working examples
- Limited numbers of literature sources
- Difficult to debug the macro script.
- The low number of tutorials

The access to documentation is not clear, and there is more than one form of documentation inside CATIA. This creates additional complication of identifying the syntax of methods and functions along with understanding the basics of the CATIA macros.

Along with the difficulty of finding the appropriate documentation, its explanation is not always entirely clear. Having only a single line for a reason for any method and short if any example.

Furthermore, incompatibility between methods and variables has been identified where the macro throughs an error or refuses to continue. Leading to running multiple tests to see which method can be applied to which variable. Ignoring the clear connection stated and showed in the documentation of individual containers (CATIA was of storing variables).

In terms of working example, some are available in the CATIA documentation if they are found, and there are a small number of models on online forums mainly the COE one. The CATIA documentation examples are limited to only one application of a method, and the application is provided inside a more complex solution. Whereas the on-forum solutions are more in dept to the specific problem, but they are extremely limited to several functionalities that the software provides. Additionally, they are based on a handful of the most popular methods and the most popular containers.

Having a small number of literature sources when there are not that many work examples is entirely expected. Only five sources on CATIA macros, two books and several forum solutions have been identified as suitable literature resource. Likewise, the number of tutorials on CATIA macros cannot suffice the support of the project. The reason for this is that those tutorials have insufficient information about the macro capabilities outside of zero level knowledge like “Hello world” message. With such an example, it can be understood that the available tutorials have low or zero explanation on automation and complex tasks execution.

It is necessary to be consistent while scripting macros in CATIA. A small error while scripting would not allow the implementation of the script. Particular attention should be given to the order and format of the script. It is recommended to keep the code clean and tidy. Only add necessary coding functions should be added in script. This would avoid or neglect the process of debugging a CATIA macro script.

All the above complications have significantly delayed the progress on the project.

Comparison

After considering both the solutions, it can be concluded that they initially take the same approach of imputing the x, y, z coordinates and defining a point. Solution 1 is more simple in terms of putting the coordinates as when the user runs the macro. A parameter box shows up in which the user can input the required fields without going inside the code whereas in solution 2 the user has to edit some commands line in the code for the required fields making it complex in terms of inputting the data.

Solution 1 can only extrude a channel on the face of the CAD model. In contrast, solution 2 can extrude channel inside the geometry as solution 2 uses a plane equation approach for generating a reference plane with the points. In comparison, Solution 1 does not use such a plane. As Solution 2 uses a plane equation, therefore multiple extrusions are possible at a single time. Still, in Solution 1, only single extrusion can be done at a time, hence, increasing the time for multiple extrusion. Increased time also means that more time to simulate on ANSYS for analyzing the data of heat transfer properties.

Therefore, after comparing both the solutions, it was concluded that Solution 2 was more suitable for this project. Due to limited time during the global pandemic and lack of resources, it was challenging to include the certain steps from Solution 1 into solution 2 to optimize the final solution meeting all the aims and objectives of this project.

In the next section, a summary of the project and the main final recommendations have been presented.

5.0 Conclusion & Future Recommendations

The current state of the project does not meet all the objectives. Those objectives are the missing GUI that could have been ignored if the application could be run from within python. The reason for that is that the client is using python as the main tool for the CFD simulations and data extraction. Unfortunately, that information was shared at a later stage of the project where changes from CATIA macro to python were not assigned and inefficient in terms of the project delivery date. Although, A graphical user interface has been created, which works fine for single-channel extrusion. The coding method can be further used to develop a more valid user form for multiple channel extrusion. As the process of coding is time-consuming, and it is next to impossible to write code without any reference sources. None of the experts in the field of CAD designing has made a code for developing a new complex system which can be used as a reference for completing this method. Information on the diameter of the extrusion tube was discussed with the client. However, the path of the channel is unclear as no Cartesian coordinates were provided. The client was also only able to share one geometrical model, which was a .stp file; hence it tough to understand the modelling process of the 3D mould model. The code created by the group allows the user to change the diameter of the extrusion channel in a simple block. A literature review conducted on the automation in CAD geometry only highlights the methods used in the field. The experts in fields are only able to create a macro code using the recording function which is available in CATIA. With more time, a better understanding of the CATIA software and linking multiple software's together, this task might be possible.

Both scripts created by the group are user friendly. Also, the scripts can create an extrusion channel. If the group had more time together, then it would have been possible to link the code to the 3D part given by the client. This then would have helped the group to meet the final and most essential objective. That is the conformal cooling objective, e.g. generating channels that are conformal in part. The missing component of this objective is the extraction method used to find the points that represent the conformal channel. Those points are a representation of the geometrical values in 3D space of a chosen surface. The conformal points values are calculated using the formula below:

$$\text{Conformal point} = \text{surface point} - (2*\text{diameter}+0.5*\text{diameter})$$

where the 0.5*diameter value represents the distance to the central point of the channel. This extraction can be said to create a sub-surface like a representation of the chosen surface positioned using the equation above.

Future recommendation

As state above the application is not fully completed to the client standards and needs; hence the following point represents the room for improvement and the future recommendations:

- Call the code from python and remove the need of the assembly method.
- Extract the conformal surface points and use them to call the channel generation method with two dataset inputs at a time.

Incorporating the assembly method from the macro scripts is useful for the client because it bypasses the need for a GUI and will significantly improve the computational speed of the created inner layout. The components that will be required to exist in the python version of the assembly is the part selection and opening in an edit mode. It needs to be followed by a function or a method that extracts the points describing the inner layout, e.g. the channel's position. Using those points, the channel generation macro can be called with two dataset inputs at a time as described above. All of the above components can be written using the pycatia library that is available and can be added using pip to the currently available python libraries on the used machine (laptop, desktop, workstation, etc.).

The client suggested the pycatia library at the final client meeting as the main tool to write the application. But on a previous stage of the project, the library was firmly studied, and it was decided that it does not have all the methods and functionalities required for developing the application from start to end. Because of that, only macro developing method was used.

In terms of the extraction of the conformal points, only the method that extracts their position is required. Due to the time constraints and insufficient literature sources and appropriate documentation, this objective could not be met. The process that needs to be used to classify the channel generator stated above is by using 2-dataset input at a time. Because this is the only method that the application supports. If the current dataset used represents the complete inner layout of the cooling system a for loop or a recursive method will be required to be used to call the channel generation macro.

References

- Ziethen, D., and Brand, K. (2013) CATIA V5: Macro Programming with Visual Basic Script [online] 1st edn. New York: McGraw-Hill Publishing. available from <https://ebookcentral.proquest.com/lib/coventry/reader.action?docID=4960659> [23 April 2020]

- Gloria Del Río-Cidoncha, M., Martínez-Palacios, J. and Ortuño-Ortiz, F. (2007) 'Task Automation for Modelling Solids with Catia V5' *Aircraft Engineering And Aerospace Technology* [online] 79 (1), 53-59. available from <https://www.emerald.com/insight/content/doi/10.1108/00022660710720494/full/pdf?title=task-automation-for-modelling-solids-with-catia-v5> [23 April 2020]

- Kumbhar, R., Pawar, S., Jadhav, D. and Dhanrale, N. (2014) 'Customization of Catia V5 for Creating Different Types of Holes on Disc Wheel' *International Journal of Engineering Research & Technology (IJERT)* [online] 3 (05), available from <https://www.ijert.org/research/customization-of-catia-v5-for-creating-different-types-of-holes-on-disc-wheel-IJERTV3IS050843.pdf> [23 April 2020]

- Siddesh S. and Suresh B. S. (2015) 'Automation of Generating CAD Models' *Journal of Mechanical Engineering and Automation* [online] 5(3B), 55-58. DOI: 10.5923/c.jmea.201502.11

- Ali M. B., Azhar bin Shafie., Dharmalingam S., Daud M. A. M., Zakaria K. A. and MJA. Latif (2018) 'IMPLEMENTATION OF AUTOMATION PROCESS IN GENERATING CAD MODELFOR RIM WHEEL' *ARPN Journal of Engineering and Applied Sciences [online] 13(2), 594-598.* available from <http://www.arpnjournals.org/jeas/research_papers/rp_2018/jeas_0118_6707.pdf> [23 April 2020]

- Thermo-mechanical Design Optimization of Conformal Cooling Channels using Design of Experiments Approach. Suchana A. Jahan,Tong Wu,Yi Zhang,Jing Zhang,Andres Tovar,Hazim Elmounayri, Elsevier 2017

- CATIA V5 Macro Programming Dieter R. Ziethen, McGrawHill,

- Thermal Comparison of Conventional and Conformal Cooling Channel Designs for a Non-Constant Thickness Screw Cap. Eric Dimla, Josep Rull-Trinidad, Andres Amador Garcia-Granada, and Guillermo Reyes. ISSN 1225-9071, 2017

- Approaches in automation [available online from] https://catiatutor.com/approaches-in-automation/ [17 April 2020]

- SLM tooling for die casting with conformal cooling channels Antonio Armillotta*, Raffaello Baraggi Dipartimento di Meccanica, Politecnico di Milano Via La Masa 1, 20156 Milano, Italy

- Renishaw available online from https://www.renishaw.com/en/renishaw-enhancing-efficiency-in-manufacturing-and-healthcare--1030 [17 April 2020]

- Standard Cooling versus Conformal Cooling - Injection Molding Industry [available online from] https://www.azom.com/article.aspx?ArticleID=17145 [17 April 2020]

- Conformal Cooling available online from Conformal cooling: Why use it now

- Die Casting (Permanent Mold) Encyclopedia of Materials: Science and Technology, W.A. Butler, Elsevier, 2001

- Comprehensive Structural Integrity, M. Jolly, Elsevier 2003

- Pressure Die Casting [available online from] http://www.themetalcasting.com/pressure-die-casting.html [20 April 2020]

- Why Conformal Makes Sense [available online from] https://www.ptonline.com/articles/why-conformal-cooling-makes-ense [20 April 2020]

- New metal 3D-printed conformal cooling is a “huge step forward” for moulding industry [available online from] https://www.britishplastics.co.uk/News/new-metal-3d-printed-conformal-cooling-is-a-huge-step-forward/ [23 April 2020]

- Hot Extrusion Dies with Conformal Cooling Channels Produced by Additive Manufacturing, Ramona Hölker, Matthias Haase, Nooman Ben Khalifa, A. Erman Tekkaya, Elsevier 2015

- Mathonline (2020) Determining a Vector Given Two Points [Online] available from <http://mathonline.wikidot.com/determining-a-vector-given-two-points> (15 Mar 2020)

- Plane Equation [Online] available from <http://www.songho.ca/math/plane/plane.html> (15 Mar 2020)

- Google Books (2020) CATIA V5 Workbook Release V5-6R2013 [Online] available from <https://books.google.co.uk/books?id=KW8RAgAAQBAJ&pg=SA2-PA34&lpg=SA2-PA34&dq=how+to+get+the+last+element+from+a+select+search+catia&source= bl&ots=u8DGFQ66mA&sig=ACfU3U1aMXbwWV6GuQVPYPTI-im1voTm-

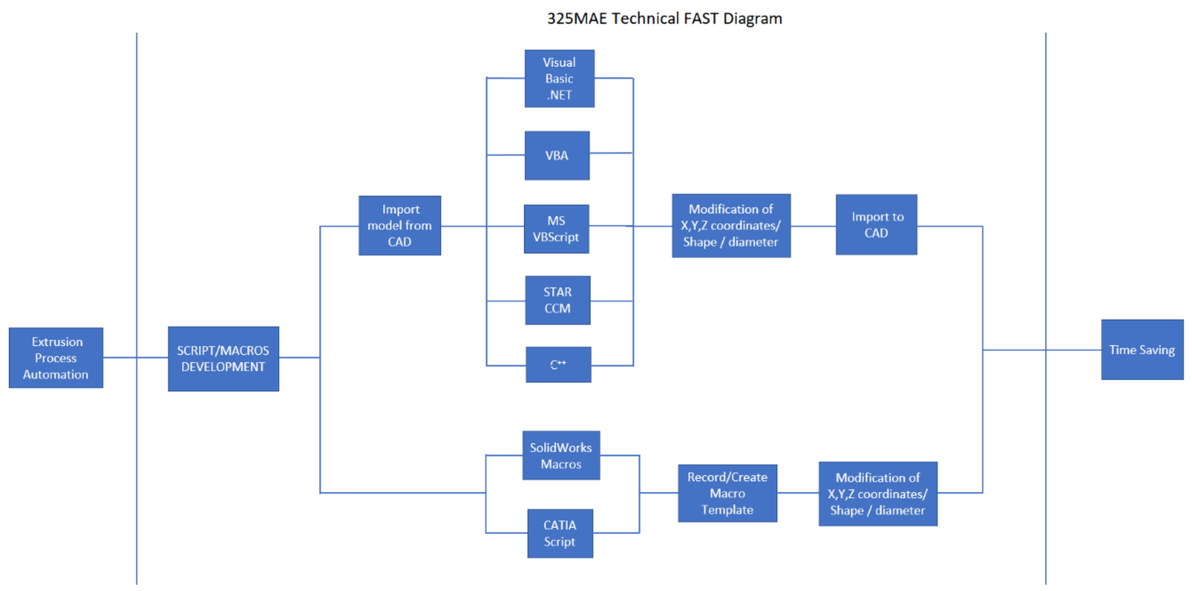
Q&hl=en&sa=X&ved=2ahUKEwichtvjlqToAhVTasAKHewlB6gQ6AEwBXoEC AoQAQ #v=onepage&q=macro&f=false> (20/03/2020)

- YouTube (2016) CATIA V5 Tutorial Beginner #9 - Sketch, Pad, Hole, Mirror, Spherical Surface Generation [Online] available from <https://www.youtube.com/watch?v=_mI7hGmvXgE> (25 May 2016)

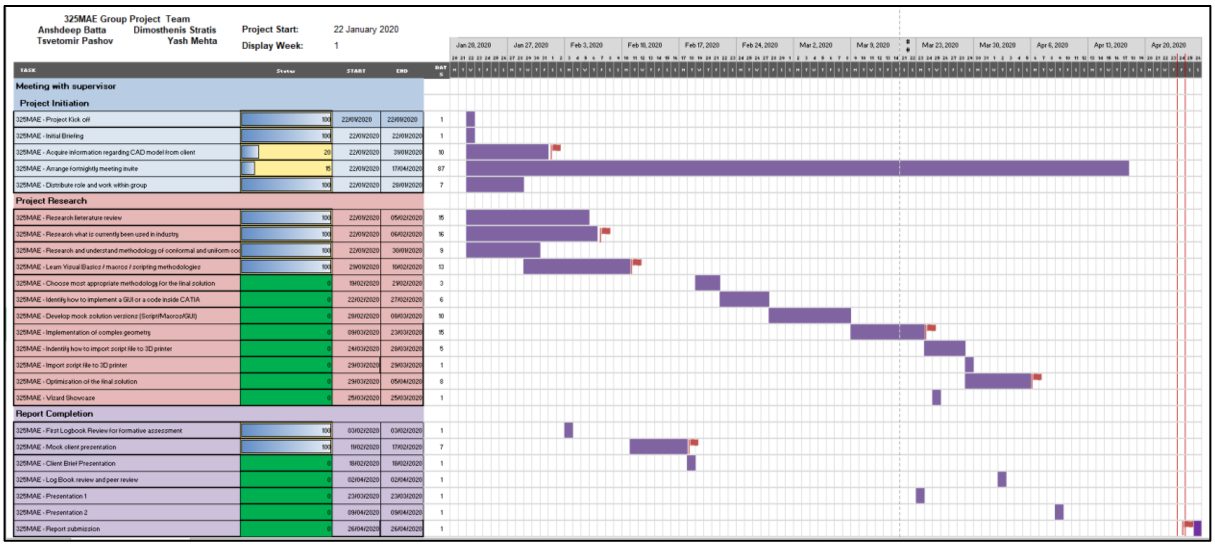
- Rand (2017) How to export specification tree [Online] available from <https://blogs.rand.com/rand3d/2017/05/how-to-export-the-specification-tree.html> (30 May 2017)

- Stack overflow (2017) Catia v5 export tree through VBA macro [Online] available from < https://stackoverflow.com/questions/46330409/catia-v5-export-tree-through-vba-macro> (20 Sep 2017)

Appendix A – Fast


325MAE Technical FAST Diagram
Extrusion Process Automation
SCRIPT/MACROS DEVELOPMENT
Import model from CAD
Visual Basic .NET
VBA
MS VBScript
STAR CCM
C++
Modification of X,Y,Z coordinates/ Shape / diameter
Import to CAD
Time Saving
SolidWorks Macros
CATIA Script
Record/Create Macro Template
Modification of X,Y,Z coordinates/ Shape / diameter

Appendix B - Gantt Chart



325MAE Group Project Team
Anshdeep Batta Dimosthenis Stratis
Tsvetomir Pashov Yash Mehta

Project Start: 22 January 2020
Display Week: 1

TASK	Status	START	END	DAYS
Meeting with supervisor				
Project Initiation				
325MAE - Project Kick off	100	22/01/2020	22/01/2020	1
325MAE - Initial Briefing	100	22/01/2020	22/01/2020	1
325MAE - Acquire information regarding CAD model from client	20	22/01/2020	31/01/2020	10
325MAE - Arrange fortnightly meeting invite	15	22/01/2020	17/04/2020	87
325MAE - Distribute role and work within group	100	22/01/2020	28/01/2020	7
Project Research				
325MAE - Research lieterature review	100	22/01/2020	05/02/2020	15
325MAE - Research what is currently been used in industry	100	22/01/2020	06/02/2020	16
325MAE - Research and understand methodology of conformal and uniform coo	100	22/01/2020	30/01/2020	9
325MAE - Learn Visual Basics / macros / scripting methodologies	100	29/01/2020	10/02/2020	13
325MAE - Choose most appropriate methodology for the final solution	0	19/02/2020	21/02/2020	3
325MAE - Identify how to implement a GUI or a code inside CATIA	0	22/02/2020	27/02/2020	6
325MAE - Develop mock solution versions (Script/Macros/GUI)	0	28/02/2020	08/03/2020	10
325MAE - Implementation of complex geometry	0	09/03/2020	23/03/2020	15
325MAE - Indentify how to import script file to 3D printer	0	24/03/2020	28/03/2020	5
325MAE - Import script file to 3D printer	0	29/03/2020	29/03/2020	1
325MAE - Optimisation of the final solution	0	29/03/2020	05/04/2020	8
325MAE - Vizard Showcase	0	25/03/2020	25/03/2020	1
Report Completion				
325MAE - First Logbook Review for formative assessment	100	03/02/2020	03/02/2020	1
325MAE - Mock client presentation	100	11/02/2020	17/02/2020	7
325MAE - Client Brief Presentation	0	18/02/2020	18/02/2020	1
325MAE - Log Book review and peer review	0	02/04/2020	02/04/2020	1
325MAE - Presentation 1	0	23/03/2020	23/03/2020	1
325MAE - Presentation 2	0	09/04/2020	09/04/2020	1
325MAE - Report submission	0	26/04/2020	26/04/2020	1

Appendix C - Pseudocode

```
Check if there is an open part
Check if there is an existing excel spreadsheet with data points
For every two consecutive datasets
        Plane equations parameters
        Generate reference point for 1st dataset
        Generate reference point for 2nd dataset
        Generate plane for 1st dataset
        Generate plane for 2nd dataset
        Create a circle on 1st plane
        Create a circle on 2nd plane
        Remove solid between the circles on 1st and 2nd plane - channel generated
        Remove spherical solid section at both ends of the channel - channel connection
Next
```

Appendix D – Code for Solution 1 in Visual Basic

The code for the first solution:

```
Sub CATMain(X As Integer, Y As Integer, Z As Integer, L As Integer, D As Integer)

Dim partDocument1 As PartDocument

Set partDocument1 = CATIA.ActiveDocument

Dim part1 As Part

Set part1 = partDocument1.Part

Dim hybridShapeFactory1 As HybridShapeFactory

Set hybridShapeFactory1 = part1.HybridShapeFactory

Dim hybridShapePointCoord1 As HybridShapePointCoord

Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(X, Y, Z)

Dim bodies1 As Bodies

Set bodies1 = part1.Bodies

Dim body1 As Body

Set body1 = bodies1.Item("PartBody")

body1.InsertHybridShape hybridShapePointCoord1

part1.InWorkObject = hybridShapePointCoord1

part1.Update

Dim sketches1 As Sketches

Set sketches1 = body1.Sketches

Dim reference1 As Reference

Set reference1 =
part1.CreateReferenceFromName("Selection_RSur:(Face:(Brp:(Pad.1;0:(Brp:(Sketch.1;4)));No
ne:();Cf11:());Pad.1_ResultOUT;Z0;G8078)")

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 point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(-Y, Z)
point2D1.ReportName = 3
Dim circle2D1 As Circle2D
Set circle2D1 = factory2D1.CreateClosedCircle(-Y, Z, D)
circle2D1.CenterPoint = point2D1
circle2D1.ReportName = 4
Dim hybridShapes1 As HybridShapes
```

```
Set hybridShapes1 = body1.HybridShapes
'Error shows up
Dim reference2 As Reference
Set reference2 = hybridShapes1.Item("Point.2")
Dim geometricElements2 As GeometricElements
Set geometricElements2 = factory2D1.CreateProjections(reference2)
Dim geometry2D1 As Geometry2D
Set geometry2D1 = geometricElements2.Item("Mark.1")
geometry2D1.Construction = True
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeRadius, reference3)
constraint1.Mode = catCstModeDrivingDimension
Dim length1 As Length
Set length1 = constraint1.Dimension
length1.Value = D
Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(geometry2D1)
Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeConcentricity, reference4, reference5)
constraint2.Mode = catCstModeDrivingDimension
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
```

```
Dim shapeFactory1 As ShapeFactory

Set shapeFactory1 = part1.ShapeFactory

Dim pocket1 As Pocket

Set pocket1 = shapeFactory1.AddNewPocket(sketch1, L)

Dim limit1 As Limit

Set limit1 = pocket1.FirstLimit

Dim length2 As Length

Set length2 = limit1.Dimension

length2.Value = L

part1.Update

End Sub
```

Appendix E – Solution 2 code

Example input

```
Const X1 = -40, Y1 = -40, Z1 = 40, Radius1 = 5, X2 = 50, Y2 = -40, Z2 = 40, Radius2 = 5


Public Sub CATMain(X1 As Integer, Y1 As Integer, Z1 As Integer, Radius1 As Integer, X2 As Integer, Y2 As Integer, Z2 As Integer, Radius2 As Integer)


'-------------------------------------------------------------------------'
' Generates the values for the START plane equation
Dim A1, B1, C1, D1
A1 = X2 - X1
B1 = Y2 - Y1
C1 = Z2 - Z1
D1 = (A1 * X1 + B1 * Y1 + C1 * Z1)

' Generates the values for the END plane equation
Dim A2, B2, C2, D2
A2 = X1 - X2
B2 = Y1 - Y2
C2 = Z1 - Z2
D2 = (A2 * X2 + B2 * Y2 + C2 * Z2)



'-------------------------------------------------------------------------'
' Main part of CATIA documentation
Dim partDocument1, part1
' Retrieve your active document
Set partDocument1 = CATIA.ActiveDocument
Set part1 = partDocument1.Part
```

```
'-----------------------------------------------------------------------'
Dim partName, bodyName, Bodies
' Extracts the active part name
partName = partDocument1.Product.Name
' Scans the collection of Bodies in a part
Set Bodies = partDocument1.Part.Bodies
'counter = Bodies.Count
bodyName = Bodies.Item(1).Name
Path = partName & "\" & bodyName

'-----------------------------------------------------------------------'
Dim hybridShapeFactory1
Set hybridShapeFactory1 = part1.HybridShapeFactory

' Generates the reference point for the START plane
Dim hybridShapePointCoord1, bodies1, body1, hybridShapePointCoord2
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(X1, Y1, Z1)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item(bodyName)
body1.InsertHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
part1.Update
' Generates the reference point for the END plane
Set hybridShapePointCoord2 = hybridShapeFactory1.AddNewPointCoord(X2, Y2, Z2)
body1.InsertHybridShape hybridShapePointCoord2
part1.InWorkObject = hybridShapePointCoord2
part1.Update
```

```
' Generates the START plane using the equation and the reference point
Dim hybridShapePlaneEquation1, reference1, hybridShapePlaneEquation2, reference2
Set hybridShapePlaneEquation1 = hybridShapeFactory1.AddNewPlaneEquation(A1, B1, C1,
D1)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointCoord1)
hybridShapePlaneEquation1.SetReferencePoint reference1
body1.InsertHybridShape hybridShapePlaneEquation1
part1.InWorkObject = hybridShapePlaneEquation1
part1.Update
' Generates the END plane using the equation and the reference point
Set hybridShapePlaneEquation2 = hybridShapeFactory1.AddNewPlaneEquation(A2, B2, C2,
D2)
Set reference2 = part1.CreateReferenceFromObject(hybridShapePointCoord2)
hybridShapePlaneEquation2.SetReferencePoint reference2
body1.InsertHybridShape hybridShapePlaneEquation2
part1.InWorkObject = hybridShapePlaneEquation2
part1.Update


'-------------------------------------------------------------------------'
' Creates a selection
Dim selection1, FirstPoint, SecondPoint
Set selection1 = partDocument1.Selection

' Traverses the selection for a specified named component
' Finds the needed point names
selection1.Search ("Name=point*,all")
selectionCount = selection1.Count2
FirstPoint = selectionCount - 1
SecondPoint = selectionCount
```

```
' Gets the name of the needed point - to be used as a reference - supports the automation of the tool

FirstPointName = selection1.Item(FirstPoint).Value.Name

SecondPointName = selection1.Item(SecondPoint).Value.Name


' Finds the needed plane names

Dim FirstPlane, SecondPlane

selection1.Search ("Name=plane*,all")

selectionCount = selection1.Count2

FirstPlane = selectionCount - 1

SecondPlane = selectionCount

FirstPlaneName = selection1.Item(FirstPlane).Value.Name

SecondPlaneName = selection1.Item(SecondPlane).Value.Name


'-------------------------------------------------------------------------'

' Generates a circle on the START planes - with specified point coordinates

Dim sketches1, hybridShapes1, reference3, sketch1, arrayOfVariantOfDouble1(8)

Dim factory2D1, geometricElements1, axis2D1, line2D1, line2D2, point2D1

Dim circle2D1, constraints1, reference4, reference5, constraint1, length1

Dim reference6, reference7, constraint2, length2, reference8, constraint3, length3

Set sketches1 = body1.Sketches

Set hybridShapes1 = body1.HybridShapes

Set reference3 = hybridShapes1.Item(FirstPlaneName)

Set sketch1 = sketches1.Add(reference3)

arrayOfVariantOfDouble1(0) = 0#

arrayOfVariantOfDouble1(1) = 0#

arrayOfVariantOfDouble1(2) = 0#

arrayOfVariantOfDouble1(3) = 0#

arrayOfVariantOfDouble1(4) = 0#

arrayOfVariantOfDouble1(5) = 0#
```

```
arrayOfVariantOfDouble1(6) = 0#

arrayOfVariantOfDouble1(7) = 0#

arrayOfVariantOfDouble1(8) = 0#

Set sketch1Variant = sketch1

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1

part1.InWorkObject = sketch1

Set factory2D1 = sketch1.OpenEdition()

Set geometricElements1 = sketch1.GeometricElements

Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

Set point2D1 = factory2D1.CreatePoint(0#, 0#)

point2D1.ReportName = 3

Set circle2D1 = factory2D1.CreateClosedCircle(0#, 0#, Radius1)

circle2D1.CenterPoint = point2D1

circle2D1.ReportName = 4

Set constraints1 = sketch1.Constraints

Set reference4 = part1.CreateReferenceFromObject(point2D1)

Set reference5 = part1.CreateReferenceFromObject(line2D2)

Set constraint1 = constraints1.AddBiEltCst(catCstTypeDistance, reference4, reference5)

constraint1.Mode = catCstModeDrivingDimension

Set length1 = constraint1.Dimension

length1.Value = 0#

Set reference6 = part1.CreateReferenceFromObject(point2D1)

Set reference7 = part1.CreateReferenceFromObject(line2D1)

Set constraint2 = constraints1.AddBiEltCst(catCstTypeDistance, reference6, reference7)

constraint2.Mode = catCstModeDrivingDimension
```

```
Set length2 = constraint2.Dimension
length2.Value = 0#
Set reference8 = part1.CreateReferenceFromObject(circle2D1)
Set constraint3 = constraints1.AddMonoEltCst(catCstTypeRadius, reference8)
constraint3.Mode = catCstModeDrivingDimension
Set length3 = constraint3.Dimension
length3.Value = Radius1
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update



' Generates a circle on the END planes - with specified point coordinates
Dim reference9, sketch2, arrayOfVariantOfDouble2(8), factory2D2, geometricElements2
Dim axis2D2, line2D3, line2D4, point2D2, circle2D2, constraints2, reference10
Dim reference11, constraint4, length4, reference12, reference13, constraint5
Dim length5, reference14, constraint6, length6
Set reference9 = hybridShapes1.Item(SecondPlaneName)
Set sketch2 = sketches1.Add(reference9)
arrayOfVariantOfDouble2(0) = 0#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 0#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 0#
```

```
Set sketch2Variant = sketch2

sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2

part1.InWorkObject = sketch2

Set factory2D2 = sketch2.OpenEdition()

Set geometricElements2 = sketch2.GeometricElements

Set axis2D2 = geometricElements2.Item("AbsoluteAxis")

Set line2D3 = axis2D2.GetItem("HDirection")

line2D3.ReportName = 1

Set line2D4 = axis2D2.GetItem("VDirection")

line2D4.ReportName = 2

Set point2D2 = factory2D2.CreatePoint(0#, 0#)

point2D2.ReportName = 3

Set circle2D2 = factory2D2.CreateClosedCircle(0#, 0#, Radius2)

circle2D2.CenterPoint = point2D2

circle2D2.ReportName = 4

Set constraints2 = sketch2.Constraints

Set reference10 = part1.CreateReferenceFromObject(point2D2)

Set reference11 = part1.CreateReferenceFromObject(line2D4)

Set constraint4 = constraints2.AddBiEltCst(catCstTypeDistance, reference10, reference11)

constraint4.Mode = catCstModeDrivingDimension

Set length4 = constraint4.Dimension

length4.Value = 0#

Set reference12 = part1.CreateReferenceFromObject(point2D2)

Set reference13 = part1.CreateReferenceFromObject(line2D3)

Set constraint5 = constraints2.AddBiEltCst(catCstTypeDistance, reference12, reference13)

constraint5.Mode = catCstModeDrivingDimension

Set length5 = constraint5.Dimension

length5.Value = 0#

Set reference14 = part1.CreateReferenceFromObject(circle2D2)
Set constraint6 = constraints2.AddMonoEltCst(catCstTypeRadius, reference14)

constraint6.Mode = catCstModeDrivingDimension

Set length6 = constraint6.Dimension

length6.Value = Radius2

sketch2.CloseEdition

part1.InWorkObject = sketch2

part1.Update



'-----------------------------------------------------------------------'

' Generates an extremum for START plane

Dim hybridShapeDirection1, reference15, hybridShapeExtremum1

Set hybridShapeDirection1 = hybridShapeFactory1.AddNewDirectionByCoord(1#, 2#, 3#)

Set reference15 = part1.CreateReferenceFromObject(sketch1)

Set hybridShapeExtremum1 = hybridShapeFactory1.AddNewExtremum(reference15,
hybridShapeDirection1, 1)

body1.InsertHybridShape hybridShapeExtremum1

part1.InWorkObject = hybridShapeExtremum1

part1.Update

' Generates an extremum for END plane

Dim hybridShapeDirection2, reference16, hybridShapeExtremum2

Set hybridShapeDirection2 = hybridShapeFactory1.AddNewDirectionByCoord(1#, 2#, 3#)

Set reference16 = part1.CreateReferenceFromObject(sketch2)

Set hybridShapeExtremum2 = hybridShapeFactory1.AddNewExtremum(reference16,
hybridShapeDirection2, 1)

body1.InsertHybridShape hybridShapeExtremum2

part1.InWorkObject = hybridShapeExtremum2

part1.Update
```

```
'-------------------------------------------------------------------------'

' Removes the solid that is between the circles generated on the START and END planes

Dim shapeFactory1, loft1, hybridShapeLoft1, reference17, reference18, reference19,
reference20

Set shapeFactory1 = part1.ShapeFactory

Set loft1 = shapeFactory1.AddNewRemovedLoft()

Set hybridShapeLoft1 = loft1.HybridShape

hybridShapeLoft1.SectionCoupling = 3

hybridShapeLoft1.Relimitation = 1

hybridShapeLoft1.CanonicalDetection = 2

Set reference17 = part1.CreateReferenceFromObject(sketch1)

Set reference18 = part1.CreateReferenceFromObject(hybridShapeExtremum1)

hybridShapeLoft1.AddSectionToLoft reference17, 1, reference18

Set reference19 = part1.CreateReferenceFromObject(sketch2)

Set reference20 = part1.CreateReferenceFromObject(hybridShapeExtremum2)

hybridShapeLoft1.AddSectionToLoft reference19, -1, reference20

part1.InWorkObject = hybridShapeLoft1

part1.Update


'-------------------------------------------------------------------------'

' Finds the needed sketch names

Dim FirstSketch, SecondSketch

selection1.Search ("Name=Sketch*,all")

selectionCount = selection1.Count2

FirstSketch = selectionCount - 1

SecondSketch = selectionCount

FirstSketchName = selection1.Item(FirstSketch).Value.Name

SecondSketchName = selection1.Item(SecondSketch).Value.Name

' Finds the needed extremum names

Dim FirstExtremum, SecondExtremum
```

```
selection1.Search ("Name=Extremum*,all")

selectionCount = selection1.Count2

FirstExtremum = selectionCount - 1

SecondExtremum = selectionCount

FirstExtremumName = selection1.Item(FirstExtremum).Value.Name

SecondExtremumName = selection1.Item(SecondExtremum).Value.Name



'----------------------------------------------------------------------'

' Hides the specified points

HiddingObjects "Point", FirstPointName, bodyName

HiddingObjects "Point", SecondPointName, bodyName

' Hides the specified planes

HiddingObjects "Plane", FirstPlaneName, bodyName

HiddingObjects "Plane", SecondPlaneName, bodyName

' Hides the specified sketches

HiddingObjects "Sketch", FirstSketchName, bodyName

HiddingObjects "Sketch", SecondSketchName, bodyName

' Hides the specified extremums

HiddingObjects "Extremum", FirstExtremumName, bodyName

HiddingObjects "Extremum", SecondExtremumName, bodyName



'----------------------------------------------------------------------'

' Generating the Groove solid removal

Dim sketch3, arrayOfVariantOfDouble3(8), factory2D3, geometricElements3, axis2D3,
line2D5

Dim line2D6, point2D3, point2D4, point2D5, circle2D3, constraints3, reference21,
reference22

Dim constraint7, length7, reference23, reference24, constraint8, length8
Dim reference25, constraint9, length9, reference26, reference27, constraint10

Set sketch3 = sketches1.Add(reference3)

arrayOfVariantOfDouble3(0) = 0#

arrayOfVariantOfDouble3(1) = 0#

arrayOfVariantOfDouble3(2) = 0#

arrayOfVariantOfDouble3(3) = 0#

arrayOfVariantOfDouble3(4) = 0#

arrayOfVariantOfDouble3(5) = 0#

arrayOfVariantOfDouble3(6) = 0#

arrayOfVariantOfDouble3(7) = 0#

arrayOfVariantOfDouble3(8) = 0#

Set sketch3Variant = sketch3

sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3

part1.InWorkObject = sketch3

Set factory2D3 = sketch3.OpenEdition()

Set geometricElements3 = sketch3.GeometricElements

Set axis2D3 = geometricElements3.Item("AbsoluteAxis")

Set line2D5 = axis2D3.GetItem("HDirection")

line2D5.ReportName = 2

Set line2D6 = axis2D3.GetItem("VDirection")

line2D6.ReportName = 3

Set point2D3 = factory2D3.CreatePoint(0#, 0#)

point2D3.ReportName = 4

Set point2D4 = factory2D3.CreatePoint(-Radius1, 0#)

point2D4.ReportName = 5

Set point2D5 = factory2D3.CreatePoint(Radius1, 0#)

point2D5.ReportName = 6

Set circle2D3 = factory2D3.CreateCircle(0#, 0#, Radius1, 3.141593, 6.283185)

circle2D3.CenterPoint = point2D3
```

```
circle2D3.ReportName = 7

circle2D3.StartPoint = point2D4

circle2D3.EndPoint = point2D5

Set constraints3 = sketch3.Constraints

Set reference21 = part1.CreateReferenceFromObject(point2D3)

Set reference22 = part1.CreateReferenceFromObject(line2D6)

Set constraint7 = constraints3.AddBiEltCst(catCstTypeDistance, reference21, reference22)

constraint7.Mode = catCstModeDrivingDimension

Set length7 = constraint7.Dimension

length7.Value = 0#

Set reference23 = part1.CreateReferenceFromObject(point2D3)

Set reference24 = part1.CreateReferenceFromObject(line2D5)

Set constraint8 = constraints3.AddBiEltCst(catCstTypeDistance, reference23, reference24)

constraint8.Mode = catCstModeDrivingDimension

Set length8 = constraint8.Dimension

length8.Value = 0#

Set reference25 = part1.CreateReferenceFromObject(circle2D3)

Set constraint9 = constraints3.AddMonoEltCst(catCstTypeRadius, reference25)

constraint9.Mode = catCstModeDrivingDimension

Set length9 = constraint9.Dimension

length9.Value = Radius1

Set reference26 = part1.CreateReferenceFromObject(point2D4)

Set reference27 = part1.CreateReferenceFromObject(line2D5)

Set constraint10 = constraints3.AddBiEltCst(catCstTypeOn, reference26, reference27)

constraint10.Mode = catCstModeDrivingDimension

sketch3.CloseEdition

part1.InWorkObject = sketch3

part1.Update
```

```
Dim sketch4, arrayOfVariantOfDouble4(8), factory2D4
Dim geometricElements4, axis2D4, line2D7, line2D8, point2D6, point2D7, line2D9
Dim constraints4, reference28, reference29, constraint11, length10
Dim reference30, reference31, constraint12, length11, reference32, reference33
Dim constraint13, length12, reference34, reference35, constraint14, length13
Set sketch4 = sketches1.Add(reference3)
arrayOfVariantOfDouble4(0) = 0#
arrayOfVariantOfDouble4(1) = 0#
arrayOfVariantOfDouble4(2) = 0#
arrayOfVariantOfDouble4(3) = 0#
arrayOfVariantOfDouble4(4) = 0#
arrayOfVariantOfDouble4(5) = 0#
arrayOfVariantOfDouble4(6) = 0#
arrayOfVariantOfDouble4(7) = 0#
arrayOfVariantOfDouble4(8) = 0#
Set sketch4Variant = sketch4
sketch4Variant.SetAbsoluteAxisData arrayOfVariantOfDouble4
part1.InWorkObject = sketch4
Set factory2D4 = sketch4.OpenEdition()
Set geometricElements4 = sketch4.GeometricElements
Set axis2D4 = geometricElements4.Item("AbsoluteAxis")
Set line2D7 = axis2D4.GetItem("HDirection")
line2D7.ReportName = 1
Set line2D8 = axis2D4.GetItem("VDirection")
line2D8.ReportName = 2
Set point2D6 = factory2D4.CreatePoint(-Radius1, 0#)
point2D6.ReportName = 3
Set point2D7 = factory2D4.CreatePoint(Radius1, 0#)
```

```
point2D7.ReportName = 4
Set line2D9 = factory2D4.CreateLine(-Radius1, 0#, Radius1, 0#)
line2D9.ReportName = 5
line2D9.StartPoint = point2D6
line2D9.EndPoint = point2D7
Set constraints4 = sketch4.Constraints
Set reference28 = part1.CreateReferenceFromObject(point2D6)
Set reference29 = part1.CreateReferenceFromObject(line2D8)
Set constraint11 = constraints4.AddBiEltCst(catCstTypeDistance, reference28, reference29)
constraint11.Mode = catCstModeDrivingDimension
Set length10 = constraint11.Dimension
length10.Value = Radius1
Set reference30 = part1.CreateReferenceFromObject(point2D6)
Set reference31 = part1.CreateReferenceFromObject(line2D7)
Set constraint12 = constraints4.AddBiEltCst(catCstTypeDistance, reference30, reference31)
constraint12.Mode = catCstModeDrivingDimension
Set length11 = constraint12.Dimension
length11.Value = 0#
Set reference32 = part1.CreateReferenceFromObject(point2D7)
Set reference33 = part1.CreateReferenceFromObject(line2D8)
Set constraint13 = constraints4.AddBiEltCst(catCstTypeDistance, reference32, reference33)
constraint13.Mode = catCstModeDrivingDimension
Set length12 = constraint13.Dimension
length12.Value = Radius1
Set reference34 = part1.CreateReferenceFromObject(point2D7)
Set reference35 = part1.CreateReferenceFromObject(line2D7)
Set constraint14 = constraints4.AddBiEltCst(catCstTypeDistance, reference34, reference35)
constraint14.Mode = catCstModeDrivingDimension
Set length13 = constraint14.Dimension
```

```
length13.Value = 0#

sketch4.CloseEdition

part1.InWorkObject = sketch4

part1.Update



Dim reference36, groove1, parameters1

Set reference36 = part1.CreateReferenceFromName("")

Set groove1 = shapeFactory1.AddNewGrooveFromRef(reference36)

Set parameters1 = part1.Parameters


' Finding the name of the last groove

Dim FinalGroove

selection1.Search ("Name=Groove*,all")

FinalGroove = selection1.Count2

GrooveName = selection1.Item(FinalGroove).Value.Name



Dim length14, parameters2, length15, reference37

Set length14 = parameters1.Item(Path & "\" & GrooveName & "\ThickThin1")

length14.Value = 1#

Set parameters2 = part1.Parameters

Set length15 = parameters2.Item(Path & "\" & GrooveName & "\ThickThin2")

length15.Value = 0#

Set reference37 = part1.CreateReferenceFromObject(sketch3)

groove1.SetProfileElement reference37


' Finding the name of the last sketch

Dim FinalSketch
```

```
selection1.Search ("Name=Sketch*,all")

selectionCountSketch = selection1.Count2

FinalSketch = selectionCountSketch

FinalSketchName = selection1.Item(FinalSketch).Value.Name


Dim reference38

Set reference38 = part1.CreateReferenceFromBRepName("WireREdge:(Wire:(Brp:(" &
FinalSketchName &
";5);None:(Limits1:();Limits2:());Cf11:());WithTemporaryBody;WithoutBuildError;WithSelectin
gFeatureSupport;MFBRepVersion_CXR15)", sketch4)

groove1.RevoluteAxis = reference38

part1.Update


' Hidding the last sketch

HiddingObjects "Sketch", FinalSketchName, bodyName



'-----------------------------------------------------------------------'

Dim sketch5 As Sketch

Set sketch5 = sketches1.Add(reference9)

Dim arrayOfVariantOfDouble5(8)

arrayOfVariantOfDouble5(0) = 0#

arrayOfVariantOfDouble5(1) = 0#

arrayOfVariantOfDouble5(2) = 0#

arrayOfVariantOfDouble5(3) = 0#

arrayOfVariantOfDouble5(4) = 0#

arrayOfVariantOfDouble5(5) = 0#

arrayOfVariantOfDouble5(6) = 0#

arrayOfVariantOfDouble5(7) = 0#

arrayOfVariantOfDouble5(8) = 0#
```

```
Set sketch5Variant = sketch5

sketch5Variant.SetAbsoluteAxisData arrayOfVariantOfDouble5

part1.InWorkObject = sketch5

Dim factory2D5 As Factory2D

Set factory2D5 = sketch5.OpenEdition()

Dim geometricElements5 As GeometricElements

Set geometricElements5 = sketch5.GeometricElements

Dim axis2D5 As Axis2D

Set axis2D5 = geometricElements5.Item("AbsoluteAxis")

Dim line2D10 As Line2D

Set line2D10 = axis2D5.GetItem("HDirection")

line2D10.ReportName = 2

Dim line2D11 As Line2D

Set line2D11 = axis2D5.GetItem("VDirection")

line2D11.ReportName = 3

Dim point2D8 As Point2D

Set point2D8 = factory2D5.CreatePoint(0#, 0#)

point2D8.ReportName = 4

Dim point2D9 As Point2D

Set point2D9 = factory2D5.CreatePoint(-Radius2, 0#)

point2D9.ReportName = 5

Dim point2D10 As Point2D

Set point2D10 = factory2D5.CreatePoint(Radius2, 0#)

point2D10.ReportName = 6

Dim circle2D4 As Circle2D

Set circle2D4 = factory2D5.CreateCircle(0#, 0#, Radius2, 3.141593, 6.283185)

circle2D4.CenterPoint = point2D8

circle2D4.ReportName = 7

circle2D4.StartPoint = point2D9
```

```
circle2D4.EndPoint = point2D10

Dim constraints5 As Constraints

Set constraints5 = sketch5.Constraints

Dim reference39 As Reference

Set reference39 = part1.CreateReferenceFromObject(point2D8)

Dim reference40 As Reference

Set reference40 = part1.CreateReferenceFromObject(line2D11)

Dim constraint15 As Constraint

Set constraint15 = constraints5.AddBiEltCst(catCstTypeDistance, reference39, reference40)

constraint15.Mode = catCstModeDrivingDimension

Dim length16 As Length

Set length16 = constraint15.Dimension

length16.Value = 0#

Dim reference41 As Reference

Set reference41 = part1.CreateReferenceFromObject(point2D8)

Dim reference42 As Reference

Set reference42 = part1.CreateReferenceFromObject(line2D10)

Dim constraint16 As Constraint

Set constraint16 = constraints5.AddBiEltCst(catCstTypeDistance, reference41, reference42)

constraint16.Mode = catCstModeDrivingDimension

Dim length17 As Length

Set length17 = constraint16.Dimension

length17.Value = 0#

Dim reference43 As Reference

Set reference43 = part1.CreateReferenceFromObject(circle2D4)

Dim constraint17 As Constraint

Set constraint17 = constraints5.AddMonoEltCst(catCstTypeRadius, reference43)

constraint17.Mode = catCstModeDrivingDimension

Dim length18 As Length
```

```
Set length18 = constraint17.Dimension
length18.Value = Radius2
Dim reference44 As Reference
Set reference44 = part1.CreateReferenceFromObject(point2D9)
Dim reference45 As Reference
Set reference45 = part1.CreateReferenceFromObject(line2D10)
Dim constraint18 As Constraint
Set constraint18 = constraints5.AddBiEltCst(catCstTypeOn, reference44, reference45)
constraint18.Mode = catCstModeDrivingDimension
sketch5.CloseEdition
part1.InWorkObject = sketch5
part1.Update

Dim sketch6, arrayOfVariantOfDouble6(8), factory2D6
Dim geometricElements6, axis2D6, line2D12, line2D13, point2D11, point2D12
Dim line2D14, constraints6, reference46, reference47, constraint19, length19
Dim reference48, reference49, constraint20, length20, reference50, reference51
Dim constraint21, length21, reference52, reference53, constraint22, length22
Set sketch6 = sketches1.Add(reference9)
arrayOfVariantOfDouble6(0) = 0#
arrayOfVariantOfDouble6(1) = 0#
arrayOfVariantOfDouble6(2) = 0#
arrayOfVariantOfDouble6(3) = 0#
arrayOfVariantOfDouble6(4) = 0#
arrayOfVariantOfDouble6(5) = 0#
arrayOfVariantOfDouble6(6) = 0#
arrayOfVariantOfDouble6(7) = 0#
arrayOfVariantOfDouble6(8) = 0#
Set sketch6Variant = sketch6
```

```
sketch6Variant.SetAbsoluteAxisData arrayOfVariantOfDouble6
part1.InWorkObject = sketch6
Set factory2D6 = sketch6.OpenEdition()
Set geometricElements6 = sketch6.GeometricElements
Set axis2D6 = geometricElements6.Item("AbsoluteAxis")
Set line2D12 = axis2D6.GetItem("HDirection")
line2D12.ReportName = 1
Set line2D13 = axis2D6.GetItem("VDirection")
line2D13.ReportName = 2
Set point2D11 = factory2D6.CreatePoint(-Radius2, 0#)
point2D11.ReportName = 3
Set point2D12 = factory2D6.CreatePoint(Radius2, 0#)
point2D12.ReportName = 4
Set line2D14 = factory2D6.CreateLine(-Radius2, 0#, Radius2, 0#)
line2D14.ReportName = 5
line2D14.StartPoint = point2D11
line2D14.EndPoint = point2D12
Set constraints6 = sketch6.Constraints
Set reference46 = part1.CreateReferenceFromObject(point2D11)
Set reference47 = part1.CreateReferenceFromObject(line2D13)
Set constraint19 = constraints6.AddBiEltCst(catCstTypeDistance, reference46, reference47)
constraint19.Mode = catCstModeDrivingDimension
Set length19 = constraint19.Dimension
length19.Value = Radius2
Set reference48 = part1.CreateReferenceFromObject(point2D11)
Set reference49 = part1.CreateReferenceFromObject(line2D12)
Set constraint20 = constraints6.AddBiEltCst(catCstTypeDistance, reference48, reference49)
constraint20.Mode = catCstModeDrivingDimension
Set length20 = constraint20.Dimension
```

```
length20.Value = 0#

Set reference50 = part1.CreateReferenceFromObject(point2D12)

Set reference51 = part1.CreateReferenceFromObject(line2D13)

Set constraint21 = constraints6.AddBiEltCst(catCstTypeDistance, reference50, reference51)

constraint21.Mode = catCstModeDrivingDimension

Set length21 = constraint21.Dimension

length21.Value = Radius2

Set reference52 = part1.CreateReferenceFromObject(point2D12)

Set reference53 = part1.CreateReferenceFromObject(line2D12)

Set constraint22 = constraints6.AddBiEltCst(catCstTypeDistance, reference52, reference53)

constraint22.Mode = catCstModeDrivingDimension

Set length22 = constraint22.Dimension

length22.Value = 0#

sketch6.CloseEdition

part1.InWorkObject = sketch6

part1.Update



Dim reference54, groove2, parameters3

Set reference54 = part1.CreateReferenceFromName("")

Set groove2 = shapeFactory1.AddNewGrooveFromRef(reference54)

Set parameters3 = part1.Parameters


' Finding the name of the last groove

Dim NewFinalGroove

selection1.Search ("Name=Groove*,all")

NewFinalGroove = selection1.Count2

NewGrooveName = selection1.Item(NewFinalGroove).Value.Name
```

```
Dim length23, parameters4, length24, reference55

Set length23 = parameters3.Item(Path & "\" & NewGrooveName & "\ThickThin1")

length23.Value = 1#

Set parameters4 = part1.Parameters

Set length24 = parameters4.Item(Path & "\" & NewGrooveName & "\ThickThin2")

length24.Value = 0#

Set reference55 = part1.CreateReferenceFromObject(sketch5)

groove2.SetProfileElement reference55


' Finding the name of the last sketch

Dim NewFinalSketch

selection1.Search ("Name=Sketch*,all")

NewselectionCountSketch = selection1.Count2

NewFinalSketch = NewselectionCountSketch

NewFinalSketchName = selection1.Item(NewFinalSketch).Value.Name


Dim reference56

Set reference56 = part1.CreateReferenceFromBRepName("WireREdge:(Wire:(Brp:(" &
NewFinalSketchName &
";5);None:(Limits1:();Limits2:());Cf11:());WithTemporaryBody;WithoutBuildError;WithSelectin
gFeatureSupport;MFBRepVersion_CXR15)", sketch6)

groove2.RevoluteAxis = reference56

part1.Update


' Hidding the last sketch

HiddingObjects "Sketch", NewFinalSketchName, bodyName


End Sub
```

```
' Hidding an object that is of a specific type
Sub HiddingObjects(TypeOfObject As String, ObjectName, bodyName)


Dim partDocument1, selection1, visPropertySet1, part1, bodies1, body1
Set partDocument1 = CATIA.ActiveDocument


Set selection1 = partDocument1.Selection


Set visPropertySet1 = selection1.VisProperties


Set part1 = partDocument1.Part


Set bodies1 = part1.Bodies


Set body1 = bodies1.Item(bodyName)


If TypeOfObject = "Plane" Then


   Dim hybridShapes1, hybridShapePlaneEquation1, bSTRING1
   Set hybridShapes1 = body1.HybridShapes


   Set hybridShapePlaneEquation1 = hybridShapes1.Item(ObjectName) ' Plane name


   Set hybridShapes1 = hybridShapePlaneEquation1.Parent


   bSTRING1 = hybridShapePlaneEquation1.Name


   selection1.Add hybridShapePlaneEquation1
```

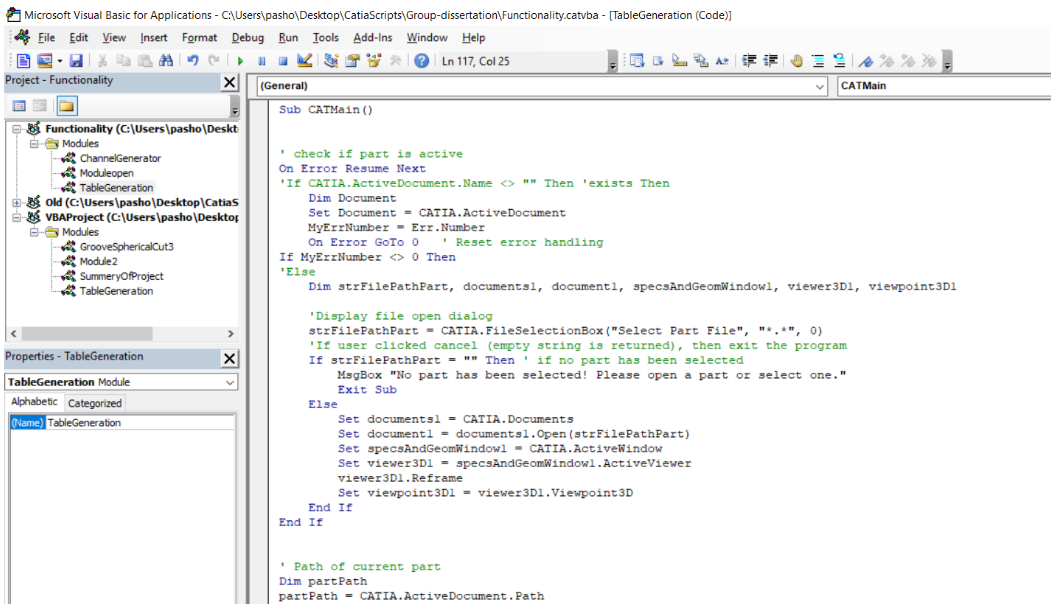
```
ElseIf TypeOfObject = "Sketch" Then

  Dim sketches1, sketch1, bSTRING2
  Set sketches1 = body1.Sketches

  Set sketch1 = sketches1.Item(ObjectName) ' Sketch name

  Set sketches1 = sketch1.Parent

  bSTRING2 = sketch1.Name

  selection1.Add sketch1

ElseIf TypeOfObject = "Extremum" Then

  Dim hybridShapes2, hybridShapeExtremum1, bSTRING3
  Set hybridShapes2 = body1.HybridShapes

  Set hybridShapeExtremum1 = hybridShapes2.Item(ObjectName) ' Extremum name

  Set hybridShapes2 = hybridShapeExtremum1.Parent

  bSTRING3 = hybridShapeExtremum1.Name

  selection1.Add hybridShapeExtremum1

ElseIf TypeOfObject = "Point" Then

  Dim hybridShapes3, hybridShapePointCoord1, bSTRING4
```

```
    Set hybridShapes3 = body1.HybridShapes

    Set hybridShapePointCoord1 = hybridShapes3.Item(ObjectName) ' Point name

    Set hybridShapes3 = hybridShapePointCoord1.Parent

    bSTRING4 = hybridShapePointCoord1.Name

    selection1.Add hybridShapePointCoord1

End If

Set visPropertySet1 = visPropertySet1.Parent

Dim bSTR2, bSTR3
bSTR2 = visPropertySet1.Name

bSTR3 = visPropertySet1.Name

visPropertySet1.SetShow 1

selection1.Clear

End Sub
```

Appendix F – Guidelines for using the application

The macro application can be run from CATIA. For the macro to be available, the library that contains the two essential macros must be added to the macro libraries.

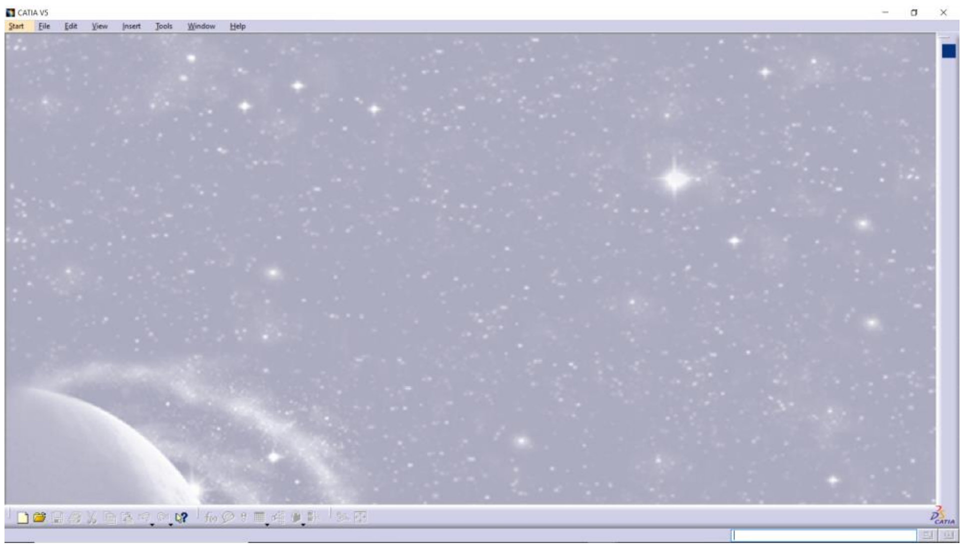
After the macros are available in the visual basics editor, the application can be run using the Tools → Macro →Macros → Functionality (macro library) →TableGenerator (assembly macro) or by pressing Alt + F11 and using the F5 method of running the same macro in the same macro library.


Microsoft Visual Basic for Applications - C:\Users\pasho\Desktop\CatiaScripts\Group-dissertation\Functionality.catvba - [TableGeneration (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 117, Col 25
Project - Functionality
Functionality (C:\Users\pasho\Deskt
Modules
ChannelGenerator
Moduleopen
TableGeneration
Old (C:\Users\pasho\Desktop\CatiaS
VBAProject (C:\Users\pasho\Desktop
Modules
GrooveSphericalCut3
Module2
SummeryOfProject
TableGeneration
Properties - TableGeneration
TableGeneration Module
Alphabetic
Categorized
(Name) TableGeneration
(General)
CATMain

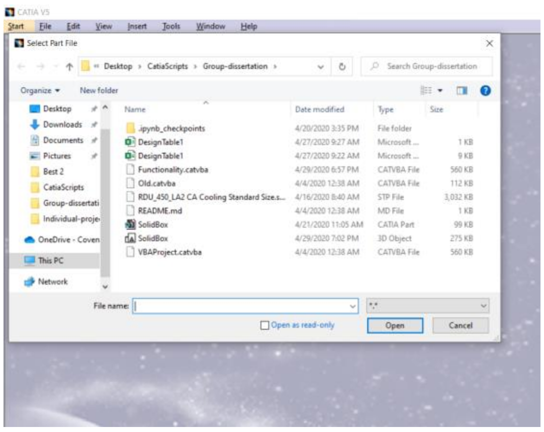
```
Sub CATMain()


' check if part is active
On Error Resume Next
'If CATIA.ActiveDocument.Name <> "" Then 'exists Then
    Dim Document
    Set Document = CATIA.ActiveDocument
    MyErrNumber = Err.Number
    On Error GoTo 0   ' Reset error handling
If MyErrNumber <> 0 Then
'Else
    Dim strFilePathPart, documents1, document1, specsAndGeomWindow1, viewer3D1, viewpoint3D1

    'Display file open dialog
    strFilePathPart = CATIA.FileSelectionBox("Select Part File", "*.*", 0)
    'If user clicked cancel (empty string is returned), then exit the program
    If strFilePathPart = "" Then ' if no part has been selected
        MsgBox "No part has been selected! Please open a part or select one."
        Exit Sub
    Else
        Set documents1 = CATIA.Documents
        Set document1 = documents1.Open(strFilePathPart)
        Set specsAndGeomWindow1 = CATIA.ActiveWindow
        Set viewer3D1 = specsAndGeomWindow1.ActiveViewer
        viewer3D1.Reframe
        Set viewpoint3D1 = viewer3D1.Viewpoint3D
    End If
End If


' Path of current part
Dim partPath
partPath = CATIA.ActiveDocument.Path
```

This is the assembly macro inside the visual basics of CATIA.

The macro can be run with or without a part being loaded.

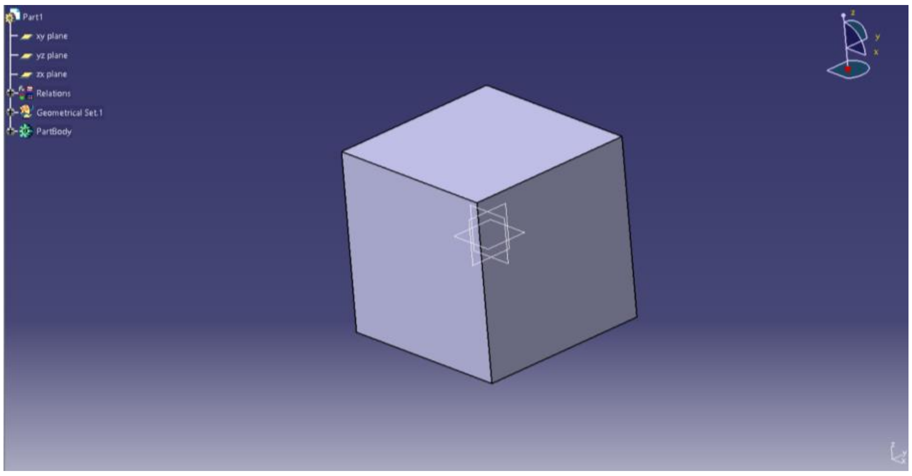


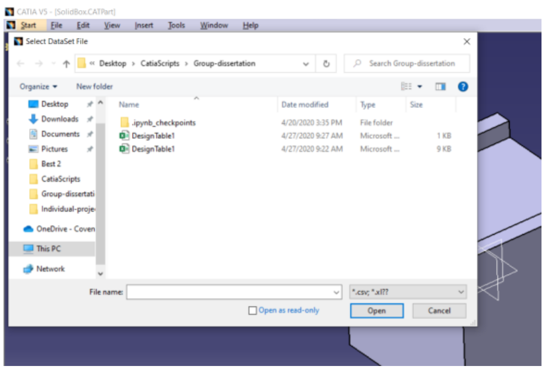
If there is not part loaded, it prompts the user to select a part that will be edited.



Name	Date modified	Type	Size
.ipynb_checkpoints	4/20/2020 3:35 PM	File folder	
DesignTable1	4/27/2020 9:27 AM	Microsoft ...	1 KB
DesignTable1	4/27/2020 9:22 AM	Microsoft ...	9 KB
Functionality.catvba	4/29/2020 6:57 PM	CATVBA File	560 KB
Old.catvba	4/4/2020 12:38 AM	CATVBA File	112 KB
RDU_450_LA2 CA Cooling Standard Size.s...	4/16/2020 8:40 AM	STP File	3,032 KB
README.md	4/4/2020 12:38 AM	MD File	1 KB
SolidBox	4/21/2020 11:05 AM	CATIA Part	99 KB
SolidBox	4/29/2020 7:02 PM	3D Object	275 KB
VBAProject.catvba	4/4/2020 12:38 AM	CATVBA File	560 KB

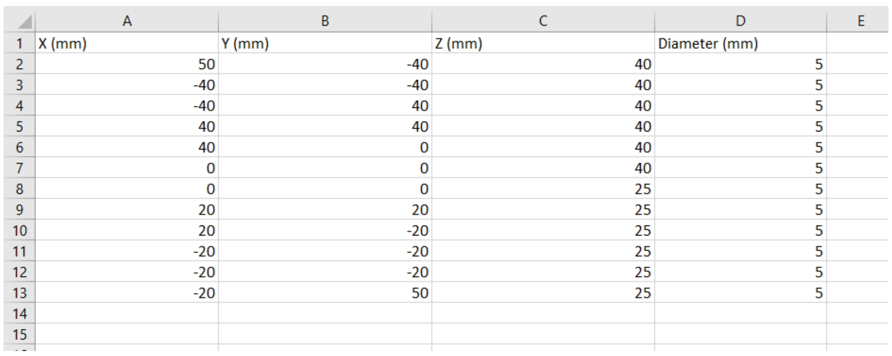
Or if there is already a part loaded the macro will do the channel generation on that part.





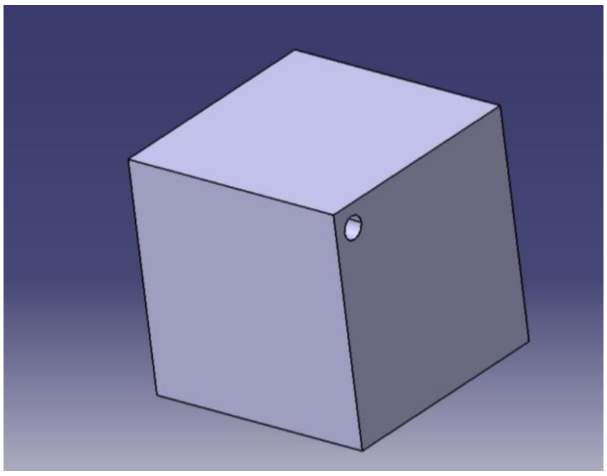
Name	Date modified	Type	Size
.ipynb_checkpoints	4/20/2020 3:35 PM	File folder	
DesignTable1	4/27/2020 9:27 AM	Microsoft ...	1 KB
DesignTable1	4/27/2020 9:22 AM	Microsoft ...	9 KB

Then it prompts the user again, but this time it requires a datasheet that contains the positions of the channels.

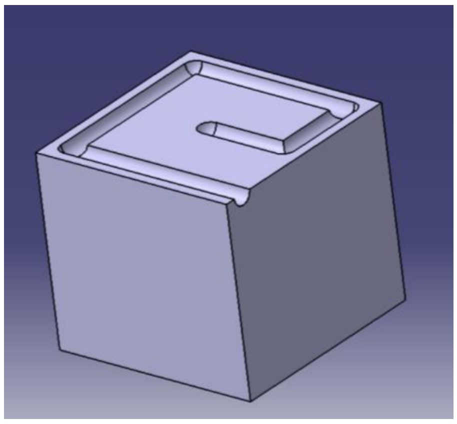


	A	B	C	D	E
1	X (mm)	Y (mm)	Z (mm)	Diameter (mm)	
2	50	-40	40	5	
3	-40	-40	40	5	
4	-40	40	40	5	
5	40	40	40	5	
6	40	0	40	5	
7	0	0	40	5	
8	0	0	25	5	
9	20	20	25	5	
10	20	-20	25	5	
11	-20	-20	25	5	
12	-20	-20	25	5	
13	-20	50	25	5	
14					
15					

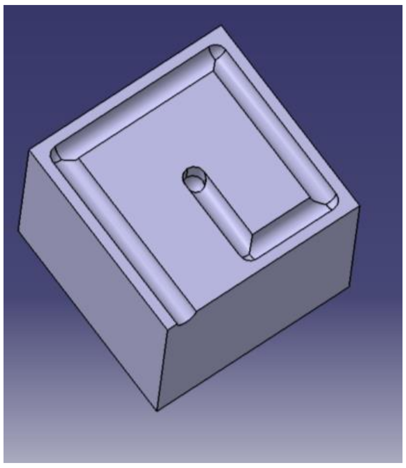
This is a simple representation of what the expected datasheet is.



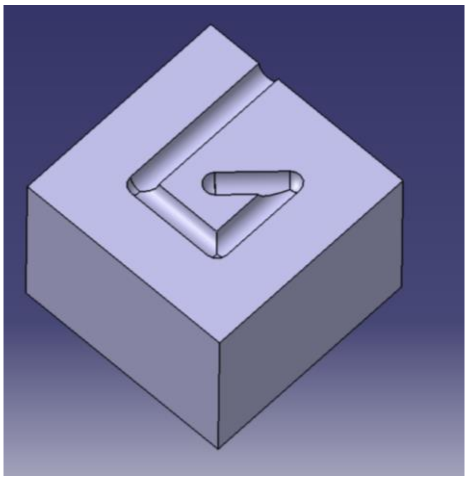
Then it starts to generate the channels



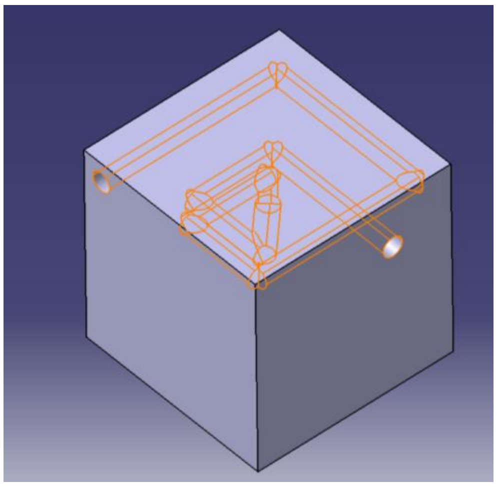
The thickness of the top surface of the box is reduced to show the channel that is generated.



Showing a channel generated vertical downwards.



Further reduction of the top surface thickness.



The final representation of the internal layout can be seen.