

Trabajo Fin de Grado

Grado en Ingeniería Aeroespacial

Desarrollo de una herramienta académica para la generación y distribución de interiores orientada al diseño de aeronaves basada en CATIA

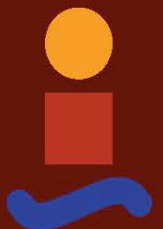
Autor: Fernando Ortega Mesas

Tutores: Sergio Esteban Roncero

Cristina Torrecillas Lozano

Dep. de Ingeniería Aeroespacial y Mecánica de Fluidos
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2016



Proyecto Fin de Grado
Grado en Ingeniería Aeroespacial

**Desarrollo de una herramienta académica para la
generación y distribución de interiores orientada al
diseño de aeronaves basada en CATIA**

Autor:

Fernando Ortega Mesas

Tutores:

Sergio Esteban Roncero

Profesor contratado Doctor

Cristina Torrecillas Lozano

Profesora contratada Doctora

Dep. de Ingeniería Aeroespacial y Mecánica de Fluidos

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Grado: Desarrollo de una herramienta académica para la generación y distribución de interiores orientada al diseño de aeronaves basada en CATIA

Autor: Fernando Ortega Mesas

Tutores: Sergio Esteban Roncero

Cristina Torrecillas Lozano

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2016

El Secretario del Tribunal

A mi familia y amigos

A mis maestros

Resumen

El contenido del presente proyecto estudia algunos de los posibles campos de aplicación del lenguaje de macros de *Microsoft Visual Basic for Applications* (VBA) al programa de diseño informático asistido por ordenador CATIA V5.

En el primer capítulo se realiza una introducción en la que se pone de manifiesto el propósito de este trabajo, el contexto en el que se sitúa el mismo y una serie de aplicaciones con las que podría identificarse.

En los capítulos 2, 3, 4, 5 y 6, se define qué es VBA, un lenguaje de macros de *Microsoft Visual Basic*, además de explicar cómo se accede desde CATIA V5, el entorno de este, el modo en que el usuario puede desarrollar interfaces para interactuar con el programa así como los objetos más importantes del módulo *Mechanical Design* (*Sketcher*, *Part Design* y *Assembly Design*) y las herramientas que ofrecen para trabajar.

En el capítulo 7, se ha pasado a describir pormenorizadamente a todo lo que envuelve este trabajo, que consiste en una aplicación que genera piezas relacionadas con los elementos comunes que se pueden ver en la sección central del fuselaje de un avión, ya sea asientos, estantes, baño, galley, carga, etc. Se ha explicado el proceso de formado de la sección central, el pasaje, pasajero, estantes, posible mercancía, carga en la bodega, baño y galley, el desarrollo del *product* y el análisis último de los posibles contactos.

En el capítulo 8, se ofrece el manual para el usuario a partir del cual, la persona que quiera hacer uso de la aplicación, sepa como interactuar con la interfaz para llevar a cabo la función requerida.

Para terminar, en los capítulos 9, 10 se describen las conclusiones a las que se ha llegado con la realización del proyecto y las posibles líneas futuras de trabajo.

Índice general

Índice general	I
Índice de figuras	IV
Índice de tablas	VII
Índice de códigos	VIII
1. Introducción	- 1 -
1.1 Motivación y objetivos	- 1 -
1.2 Contexto	- 2 -
1.3 Estado del Arte	- 4 -
1.3.1 CEASIOM	- 4 -
1.3.2 AAA	- 5 -
1.3.3 XFLR5	- 6 -
2. El entorno de programación VBA en CATIA V5	- 7 -
2.1 CATIA V5	- 7 -
2.2 Visual Basic for Application	- 8 -
2.2.1 Entorno	- 8 -
2.2.2 Librerías de las macros	- 9 -
2.2.3 Macro Recording	- 10 -
2.3 Visual Basic Editor	- 10 -
2.4 Iniciación a la programación	- 14 -
2.4.1 Declaración de estamentos	- 15 -
2.4.2 Estamentos	- 15 -
2.4.3 Estamentos ejecutables	- 16 -
2.4.4 Funciones y subfunciones	- 16 -
2.4.5 Estructuras condicionales e iterativas	- 16 -
2.4.5.1 Condicionales	- 16 -
2.4.5.2 Iterativas	- 17 -
2.4.6 Objetos orientados a la programación	- 18 -
2.4.7 Cómo definir un objeto	- 18 -
2.5 Interfaz con el usuario	- 18 -
3. Herramientas de programación de piezas	- 22 -
3.1 Introducción	- 22 -
3.2 Arranque	- 22 -

4.	Sketcher.....	- 25 -
4.1	Crear un punto.....	- 26 -
4.2	Crear una recta	- 26 -
4.3	Crear una elipse.....	- 27 -
4.4	Restricciones	- 28 -
4.4.1	CatCstTypeDistance	- 29 -
4.4.2	CatCstTypeRadius	- 30 -
4.4.3	CatCsTypeHorizontally/Vertically.....	- 30 -
4.4.4	CatCstTypeLength.....	- 31 -
4.4.5	CatCstTypeParallelism	- 31 -
4.4.6	CatCstTypeOn	- 31 -
5.	CATPart	- 33 -
5.1.	Part Design.....	- 33 -
5.1.1	Pad	- 36 -
5.1.2	Pocket	- 37 -
5.1.3.	Rectangular Pattern	- 38 -
6.	CATProduct	- 41 -
6.1	Assembly Design	- 41 -
6.1.1	AddComponentsFromFiles.....	- 42 -
6.1.2	Fix Component	- 42 -
7.	Aplicación: entorno de programación.....	- 44 -
7.1	Ventana emergente para seleccionar la carpeta de guardado.....	- 46 -
7.2	Sección central.....	- 48 -
7.3	Pasaje	- 53 -
7.3.1	Pasaje de un pasillo	- 54 -
7.3.2	Pasaje de dos pasillos	- 59 -
7.4	Pasajero.....	- 61 -
7.5	Carga a transportar.....	- 62 -
7.5.1	Pasajeros	- 63 -
7.5.2	Mercancía	- 68 -
7.6	Carga en la bodega.....	- 69 -
7.7	Baño	- 72 -
7.8	Galley	- 75 -

7.9	Product	- 78 -
7.9.1	Análisis de resultados desde el árbol de navegación.....	- 79 -
7.9.2	Análisis de resultados a través de un archivo <i>.xml</i>	- 81 -
8.	Aplicación Diseño de Interiores: manual de usuario	- 86 -
9.	Conclusiones	- 98 -
10.	Líneas futuras	- 99 -
11.	Bibliografía	- 100 -
12.	Anexo 1: Códigos de programación.....	- 101 -

Índice de figuras

Figura 1. Interacción entre las distintas áreas de Diseño.....	- 3 -
Figura 2. Ideal planes.	- 4 -
Figura 3. Esquema de los distintos objetivos de CEASIOM.....	- 5 -
Figura 4. Logotipo de AAA.....	- 6 -
Figura 5. Definición del fuselaje en XFLR5.	- 6 -
Figura 6. Acceso a macros.....	- 9 -
Figura 7. Entorno VBA.	- 11 -
Figura 8. Project Explorer y Properties Window.	- 12 -
Figura 9. Object Browser.	- 13 -
Figura 10. Formulario de trabajo.....	- 19 -
Figura 11. Herramientas de diseño.....	- 19 -
Figura 12. Editor.....	- 20 -
Figura 13. Comandos para hacer funcionar la aplicación.	- 21 -
Figura 14. Estructura del módulo <i>Sketcher</i>	- 25 -
Figura 15. Estructura del módulo <i>Part Design</i>	- 34 -
Figura 16. Estructura interna del <i>PartDocument</i>	- 34 -
Figura 17. Estructura interna del <i>ProductDocument</i>	- 41 -
Figura 18. Elementos que van a constituir el <i>product</i>	- 44 -
Figura 19. Interfaz del programa.	- 45 -
Figura 20. Guardado I.....	- 47 -
Figura 21. Guardado II.	- 48 -
Figura 22. Formulario de la sección central.	- 49 -
Figura 23. Características de la sección central.....	- 50 -
Figura 24. Creación de las elipses concéntricas.	- 51 -
Figura 25. Suelo de la cabina.....	- 52 -
Figura 26. Sección central del fuselaje.....	- 53 -
Figura 27. Formulario correspondiente a la elección del número de clases.....	- 53 -
Figura 28. Formulario correspondiente a la elección del número de pasillos.	- 54 -
Figura 29. Formulario correspondiente al pasaje de un pasillo.....	- 55 -
Figura 30. Offset delantero.....	- 56 -
Figura 31. Pitch.	- 57 -
Figura 32. Separación entre asientos.	- 57 -

Figura 33. Anchura del pasillo.	- 58 -
Figura 34. Pasaje de un pasillo.	- 59 -
Figura 35. Formulario correspondiente al pasaje de dos pasillos.	- 59 -
Figura 36. Pasaje de dos pasillos.	- 60 -
Figura 37. Formulario correspondiente a la definición del pasajero.	- 61 -
Figura 38. Pasajero.	- 62 -
Figura 39. Formulario correspondiente a la elección del tipo de carga.	- 62 -
Figura 40. Formulario correspondiente a la elección del número de clases.	- 63 -
Figura 41. Formulario correspondiente a la distribución de los estantes.	- 63 -
Figura 42. Formulario correspondiente a la carga de un pasillo.	- 64 -
Figura 43. Porcentaje de la altura del asiento por encima del mismo.	- 65 -
Figura 44. Restricción en altura.	- 66 -
Figura 45. Hueco del pasillo.	- 67 -
Figura 46. Estantes para un pasillo.	- 67 -
Figura 47. Estantes para dos pasillos.	- 68 -
Figura 48. Formulario correspondiente a las dimensiones de la mercancía.	- 68 -
Figura 49. Mercancía.	- 69 -
Figura 50. Formulario correspondiente a la definición de la carga.	- 70 -
Figura 51. Disposición de carga I.	- 70 -
Figura 52. Disposición de carga II.	- 71 -
Figura 53. Modelo elegido.	- 71 -
Figura 54. Carga en la bodega.	- 72 -
Figura 55. Formulario correspondiente a la definición del baño.	- 72 -
Figura 56. Restricción en altura.	- 74 -
Figura 57. Restricción en anchura.	- 74 -
Figura 58. Baño.	- 75 -
Figura 59. Formulario correspondiente a la definición del galley.	- 76 -
Figura 60. Restricción en altura.	- 77 -
Figura 61. Galley.	- 77 -
Figura 62. Formulario correspondiente a la elección de <i>parts</i>	- 78 -
Figura 63. Árbol de navegación.	- 79 -
Figura 64. Ventana correspondiente a los resultados del análisis.	- 80 -
Figura 65. Ejemplo.	- 80 -
Figura 66. Abierto con navegador web I.	- 82 -

Figura 67. Abierto con navegador web II.....	- 82 -
Figura 68. Descripción de la interferencia tomada como ejemplo.....	- 83 -
Figura 69. Ejemplo completo I.....	- 84 -
Figura 70. Ejemplo completo II.....	- 84 -
Figura 71. Ejemplo completo III.....	- 85 -
Figura 72. Biblioteca actual de macros.....	- 86 -
Figura 73. Pantalla de selección de la librería de macros.....	- 87 -
Figura 74. Pantalla de selección del directorio.....	- 87 -
Figura 75. Cambio del tipo de archivo a visualizar.....	- 88 -
Figura 76. Selección del proyecto <i>.catvba</i>	- 88 -
Figura 77. Interfaz de la aplicación Diseño de Interiores.....	- 89 -
Figura 78. Características de la sección central ejemplo.....	- 90 -
Figura 79. Número de clases del ejemplo.....	- 91 -
Figura 80. Características de la primera clase.....	- 91 -
Figura 81. Características de la segunda clase.....	- 92 -
Figura 82. Estantes de la primera clase.....	- 92 -
Figura 83. Estantes de la segunda clase.....	- 93 -
Figura 84. Características del baño.....	- 93 -
Figura 85. Características del galley.....	- 94 -
Figura 86. Selección de piezas que conformen el <i>product</i>	- 95 -
Figura 87. Ejemplo realizado I.....	- 95 -
Figura 88. Ejemplo realizado II.....	- 96 -
Figura 89. Vista lateral de ejemplo realizado.....	- 96 -
Figura 90. Análisis en el navegador web I.....	- 97 -
Figura 91. Análisis en el navegador web II.....	- 97 -

Índice de tablas

Tabla 1. Conceptos de la programación.	- 15 -
Tabla 2. Diferencias entre clase y objeto.....	- 18 -
Tabla 3. Opciones de la barra de herramientas.....	- 20 -
Tabla 4. Objetos del módulo <i>Sketcher</i>	- 26 -
Tabla 5. Restricciones.....	- 29 -
Tabla 6. Objetos del <i>Part Design</i>	- 35 -
Tabla 7. Herramientas del <i>ShapeFactory</i>	- 36 -
Tabla 8. Herramientas del módulo <i>Assembly Design</i>	- 42 -

Índice de códigos

Código 1. Arranque de documentos.	- 22 -
Código 2. <i>Bodies</i>	- 23 -
Código 3. <i>Sketches</i>	- 23 -
Código 4. Vector de coordenadas.	- 23 -
Código 5. Objeto <i>Factory2D</i>	- 24 -
Código 6. Creación de un punto.	- 26 -
Código 7. Creación de una recta.	- 26 -
Código 8. Creación de un círculo.	- 27 -
Código 9. Creación de un arco de circunferencia.	- 28 -
Código 10. <i>CatCstTypeDistance</i>	- 30 -
Código 11. <i>CatCstTypeRadius</i>	- 30 -
Código 12. <i>CatCstTypeHorizontally/Vertically</i>	- 30 -
Código 13. <i>CatCstTypeLength</i>	- 31 -
Código 14. <i>CatCstTypeParallelism</i>	- 31 -
Código 15. <i>CatCstTypeOn</i>	- 32 -
Código 16. Definición del objeto <i>ShapeFactory</i>	- 36 -
Código 17. <i>Pad</i>	- 36 -
Código 18. Referencia del <i>pad</i> 1.	- 37 -
Código 19. Referencia del <i>pad</i> 2.	- 37 -
Código 20. <i>Pocket</i>	- 38 -
Código 21. <i>Rectangular Pattern</i>	- 39 -
Código 22. <i>AddComponentsFromFiles</i>	- 42 -
Código 23. <i>Fix Component</i>	- 43 -
Código 24. Llamada a los <i>Userforms</i> con un <i>CommandButton</i>	- 45 -
Código 25. Ventana emergente I.	- 46 -
Código 26. Guardado I.	- 47 -
Código 27. Ventana emergente II.	- 48 -
Código 28. Bucle empleado para la caracterización de las clases.	- 54 -
Código 29. Bloque que permite exportar el análisis en archivo <i>.xml</i>	- 81 -

1. Introducción

CATIA V5 (*Computer Aided Three Dimensional Interactive Application*) es un programa que proporciona nuevas soluciones de diseño y fabricación y está ocupando un puesto de privilegio en el modelado sólido dentro del ámbito profesional siendo empleado en sectores tan importantes como el aeronáutico o la automoción.

En vista de las numerosas y tan versátiles aplicaciones del software, desde el departamento de Ingeniería Aeroespacial y Mecánica de Fluidos se decidió integrarlo en el proceso de ingeniería concurrente, tan importante en la asignatura de Cálculo de Aeronaves y así poder seguir la línea de trabajo ya establecida a la hora de desarrollar una serie de herramientas que permitan un mejor empleo del tiempo en tareas de optimización del diseño y no centrarlo en tareas superfluas de aprendizaje previo y formulación de ecuaciones.

1.1 Motivación y objetivos

El inicio del desarrollo de una aeronave requiere un gran desembolso económico que en la mayoría de los casos se ve incrementado por errores cometidos en las distintas fases del diseño. Cuanto más avanzado se encuentra el estado del proyecto, más caro resulta resolver dichos errores por lo que contar con unos resultados fruto de estudios preliminares que sean lo más cercanos a la realidad posible es algo muy interesante aunque no en todos los casos alcanzable. Es aquí donde surge la necesidad de contar con herramientas que nos permitan alcanzar una buena precisión en las estimaciones.

El objetivo del proyecto es el desarrollo de una herramienta que permita a los alumnos que la usen, estudiar la viabilidad en cuestión de espacio de la disposición de los elementos comunes que encontramos en el interior del avión mediante modelado 3D, a través de la interacción y aplicación del lenguaje de programación de macros VBA, sin necesidad de tener un conocimiento profundo de CATIA V5.

Este propósito nace con la idea de servir como soporte al departamento de Diseño dentro de la asignatura de Cálculo de Aeronaves impartida en el cuarto curso del grado en Ingeniería Aeroespacial. Cabe recordar que los alumnos adquieren a lo largo de su formación una serie de herramientas (piezas del rompecabezas), pero no se les da las instrucciones que les permita encajar dichas piezas de tal manera que resulten en un

diseño un avión completo. Además, este departamento no se había enfrentado hasta ahora con las distintas dificultades que se le presentaban a la hora de integrarse en la ingeniería concurrente (también llamada colectiva), muy destacada en esta asignatura. Se espera que éste sea el primero de muchos pasos. Para ello, en este trabajo se va a desarrollar un software con el que el usuario pueda obtener una representación gráfica automática de la disposición de esos elementos comunes y verificar que es posible, y en caso de no serlo, poder realimentar el proceso.

Por último, el alcance de este trabajo de fin de grado en Ingeniería Aeroespacial, es familiarizarse y controlar los conocimientos de programación necesarios para elaborar ésta o cualquier otra aplicación y facilitar a cualquier usuario el acceso al software realizado para cumplir sus necesidades.

1.2 Contexto

A diferencia del resto de materias impartidas en este grado, el método de evaluación de la asignatura de Cálculo de Aeronaves sigue la filosofía *Project Based Learning* (Aprendizaje Basado en Proyectos). En grupos de entorno a 10 personas que se constituyen como empresas ficticias del sector aeronáutico, los alumnos deben desarrollar un diseño conceptual de aeronave que dé respuesta al Request For Proposal (RFP) presentado por el profesor. Para llevar a cabo esta tarea, cada grupo debe llevar a cabo una estructuración para quedar repartidos en distintos departamentos, cada uno de los cuales se centrará un aspecto concreto del diseño. Así se puede nombrar a los departamentos de Aerodinámica, Actuaciones y Propulsión, Estabilidad y Control y Pesos y Estructuras.

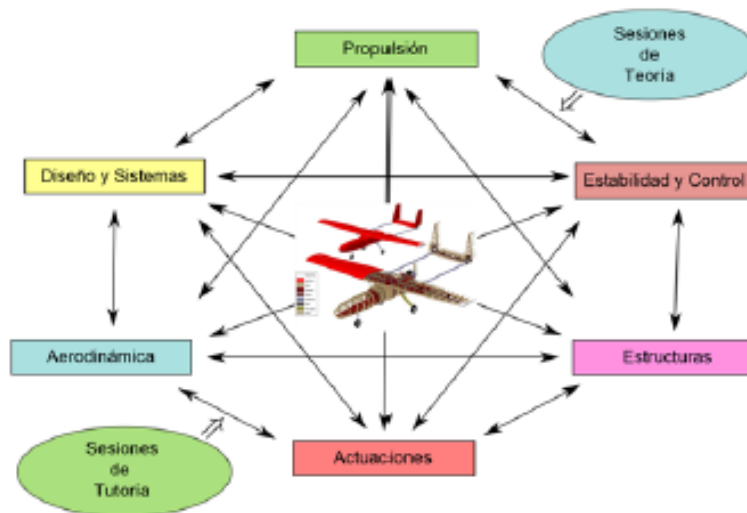


Figura 1. Interacción entre las distintas áreas de Diseño.

Desde el punto de vista de un ingeniero, el diseño completo de un avión es el fruto de un compromiso entre el conocimiento, experiencia y deseo de los muchos ingenieros que forman los distintos grupos de diseño necesarios para acometer dicha tarea. Resulta natural el entender que para cada uno de los grupos de diseño, pueden considerar que su área de responsabilidad es más importante que el resto de áreas necesarias para el diseño completo de un avión, lo que si no se conduce adecuadamente, puede llevar a la visión de C. W. Miller en la Figura 2, en la que describe lo que podría pasar si se dejara que cada uno de los distintos grupos de diseño se tomaran en serio el que su grupo es el más importante a la hora de acometer el diseño de un avión.



Figura 2. Ideal planes.

Para enfrentar dicho proceso, las herramientas pretenden ser una ayuda para algunos de estos departamentos, de manera que los alumnos puedan emplear su tiempo y esfuerzo en asimilar los contenidos realmente importantes de la asignatura y no en transcribir atentamente largas expresiones a código que después requerirán aún más tiempo para descubrir los posibles errores en la programación. En particular, la herramienta presentada en este documento pretende ser una ayuda para el Departamento de Diseño y Sistemas.

1.3 Estado del Arte

El cálculo de aeronaves es una disciplina con un cierto camino recorrido dentro de la industria y por ello, ya hay determinadas herramientas que realizan el trabajo de cómputo a partir de una serie de datos de diseño de la aeronave.

1.3.1 CEASIOM

CEASIOM (Computerised Environment for Aircraft Synthesis and Integrated Optimisation Methods) es un marco de trabajo que cuenta con herramientas que dan soporte a las distintas fases y aspectos del desarrollo conceptual de la aeronave.

Se podría englobar dentro de la nueva corriente de métodos de tipo computacional, los cuales a partir de principios básicos son capaces de calcular la información que se necesita, sustituyendo así a los métodos de manual basados en datos históricos y teoría semi-empírica.



Figura 3. Esquema de los distintos objetivos de CEASIOM.

1.3.2 AAA

AAA (Advanced Aircraft Analysis) es una aplicación de DARcorporation cuyo objetivo no es otro que dar soporte al proceso iterativo y único del diseño preliminar de aeronaves. Permite a los estudiantes e ingenieros de diseño tomar una configuración de aeronave desde la estimación de pesos inicial hasta los bucles abiertos y cerrados para la estabilidad dinámica y los análisis de sensibilidad finales, todo ello bajo las restricciones de normativas y de coste.

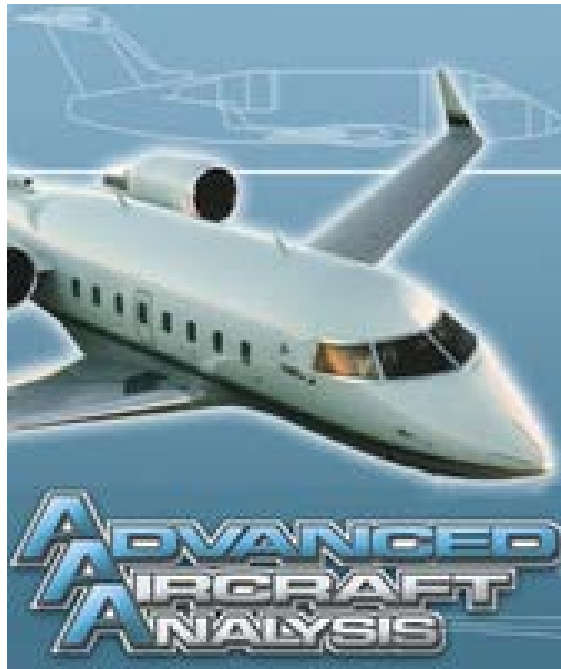


Figura 4. Logotipo de AAA.

1.3.3 XFLR5

Se trata de un software gratuito y de código abierto concebido para el diseño de fuselajes y superficies sustentadoras mediante el análisis aerodinámico y de estabilidad de los mismos. Para poder estudiar la influencia aerodinámica del fuselaje en las superficies, XFLR5 permite generar la forma del fuselaje y modificarla una serie de puntos, tal y como se muestra en la Figura 5:

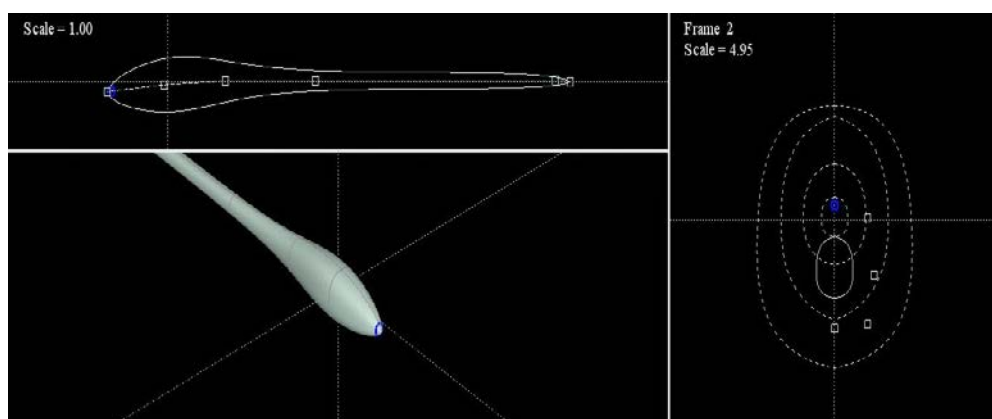


Figura 5. Definición del fuselaje en XFLR5.

2. El entorno de programación VBA en CATIA V5

Lo que se va a mostrar en este capítulo es el programa CATIA V5 y comentar grosso modo el lenguaje de programación “Visual Basic for Applications” (VBA). Una vez introducido el entorno en el que se va a trabajar, se procederá a explicar los puntos elementales de VB6, así como los fundamentos de programación de dicho lenguaje.

2.1 CATIA V5

CATIA (computer-Aided Three dimensional Interactive Application) es un programa informático de diseño, fabricación e ingeniería asistida por computadora comercial, realizado por Dassault Systèmes. CATIA es la solución líder en todo el mundo para la experiencia y el diseño de productos. Organizaciones líderes de distintos sectores la utilizan para desarrollar los productos que vemos y usamos en nuestra vida cotidiana.

Esta herramienta ofrece la posibilidad única no solo de modelar cualquier producto, sino de hacerlo en el contexto de su comportamiento en la vida real: *diseño en la era de la experiencia*. Los arquitectos de sistemas, los ingenieros, los diseñadores y todos sus colaboradores pueden definir el mundo que nos conecta, imaginarlo y darle forma.

CATIA, que se basa en la plataforma 3DEXPERIENCE de Dassault Systèmes, ofrece lo siguiente:

- Entorno de diseño social basado en una fuente única de autenticidad, al que se accede mediante potentes paneles en 3D que impulsan la inteligencia empresarial, el diseño simultáneo en tiempo real y la colaboración de todas las partes interesadas, incluidos los trabajadores.
- 3DEXPERIENCE ofrece una experiencia intuitiva con funcionalidades de modelado y simulación en 3D de primera magnitud que optimizan la eficacia de todos los usuarios.
- Se trata de una plataforma inclusiva de desarrollo de productos, que resulta cómoda de integrar con los procesos y las herramientas existentes. Esto posibilita que varias disciplinas aprovechen las eficaces e integradas aplicaciones especializadas en todas las fases del proceso de desarrollo de los productos. *Referencia [1]*.

2.2 Visual Basic for Application

Este lenguaje de programación procede del denominado *BASIC (Beginner's All-purpose Symbolic Instruction Code)* creado en 1964 en el *Dartmouth College*, como medio esencial para iniciarse en el ámbito de la programación. Modificado en sucesivas ocasiones, en 1978 se estableció el *BASIC standard*. Denominado en sus inicios como *GW-BASIC*, *QuickBASIC* posteriormente y en la actualidad *Visual Basic* adaptándose al entorno de ventanas “*Windows*” e incorporando herramientas de tipo visual como botones, listas o cuadros de texto asociadas a eventos. La primera versión de *Visual Basic* fue presentada en 1991, siendo la última la versión 6, liberada en 1998.

Automatizar tareas cotidianas, crear aplicaciones y servicios de bases de datos para el escritorio son sus utilidades principales, permitiendo el acceso a las funcionalidades de un lenguaje orientado a eventos con entrada a la API de *Windows*.

Visual Basic for Applications (VBA) es el lenguaje de macros de *Visual Basic V6*, incorporado en muchas aplicaciones de *Microsoft* y posteriormente en otras aplicaciones para ampliar la funcionalidad de las mismas. VBA incorpora las librerías y otras de objetos propias de cada software donde está incluido, además de herramientas de *Visual Basic*. La fragilidad de este lenguaje radica en que la compilación de la macro no puede realizarse si no se dispone del entorno en el que se ha desarrollado. Otra flaqueza es su falta de versatilidad para trabajar en otros sistemas operativos.

CATIA en 1998 con la versión V5 incorporó VBA a su entorno, pudiendo realizar macros en VB y en lenguaje C++, siendo aún los lenguajes de macros que se han dispuesto para su versión V6.

En el contexto del presente trabajo, las macros van a ser un pilar fundamental para el uso de la herramienta en cuanto a la generación y distribución de las piezas y el análisis de interferencias posterior en el conjunto ensamblado por parte de un usuario que, a priori, no posea unos conocimientos amplios del programa informático CATIA. *Referencia [2]*.

2.2.1 Entorno

¿Qué es una macro? Una macro consta de una serie de funciones escritas en un lenguaje de programación que agrupa una serie de comandos, los cuales permiten realizar

las operaciones requeridas automáticamente. Se usan para reducir el tiempo y la posibilidad de errores humanos cuando se realicen operaciones de forma repetitiva.

Es ilimitado el uso de macros para la automatización en el proceso de diseño. Serían ejemplos de ello la importación de puntos desde Excel a un modelo CAD 3D, la generación de geometrías de manera automática, la creación de planos de modelos 3D, etc.

Pinchando en la pestaña *Tools* desde cualquier módulo de CATIA, se accede al entorno de trabajo VBA donde se distingue la opción Macro y dentro de ésta se podrá o bien comenzar a grabar una macro o acceder a las macros ya realizadas y librerías o al editor de *Visual Basic*. Referencia [3].

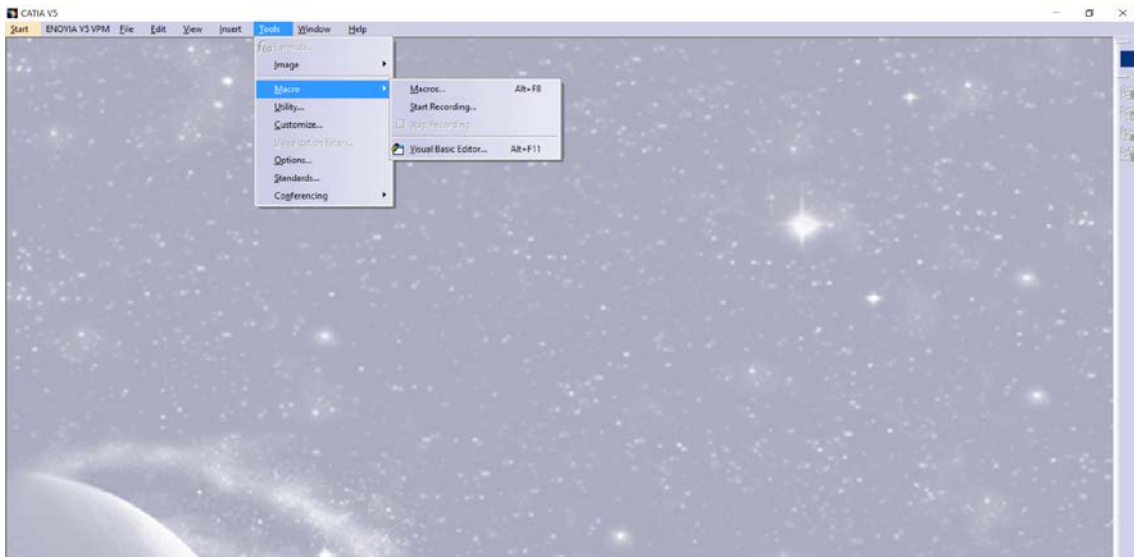


Figura 6. Acceso a macros.

2.2.2 Librerías de las macros

Las macros de CATIA son almacenadas en las librerías de macros de tres formas posibles: *Folders* (vbscript y CATScript), *Project files* (catvba) o CATParts/CATProducts. Solo una de estas tres librerías de macros puede ser usada a la vez. Para crear una nueva librería el procedimiento a usar es el siguiente:

1. Ir a *Tools* → *Macro* → *Macros*.
2. Abrir “*Macro libraries*”.
3. Asegurarse de que el tipo de librería esté cargada en “*Directories*” y luego hacer clic en “*Add existing library*”.

4. Seleccionar la carpeta donde se van a guardar los CATScripts a lo largo del proyecto.
5. Cerrar la librería de macros. En dicha librería creada deberían aparecer la lista de CATScripts que se realicen. *Referencia [3]*.

2.2.3 Macro Recording

Todas las acciones que se realicen con el ratón pueden ser grabadas, construyendo así un procedimiento para crear macros. Un método para crear macros es grabando las acciones que se realicen con el ratón. Para macros grabadas en un fichero, en un CATPart o CATProduct, los estamentos declarados se grabarán para CATScript pero no para MSVBScript. Para macros grabadas en una librería, “MS VBA” es la única opción. Varios de los aspectos a tener en cuenta si se graba una macro mediante este método son:

- No seleccionar *Workbenches* (entornos de trabajo) mientras se está grabando una macro.
- No grabar más de lo que sea absolutamente necesario.
- No usar la opción “deshacer” mientras se está grabando.
- Ser consciente y darse cuenta de la configuración de CATIA cuando se está grabando.
- Salir de los *sketches* (dibujos) antes de parar de grabar.
- Verificar cada macro una vez se haya grabado.

Terminada la grabación, se deshará todo lo realizado y se reproducirá la macro. Así se comprobará si es correcta la macro y si reproducirá la operación que se pretendía realizar.

Otro punto a tener en cuenta es la aparición de numerosas líneas de código que no son realmente necesarias, por lo que pueden eliminarse.

Tampoco aparecerán comentarios acerca de lo que se está realizando o explicando los parámetros de entrada, por lo que se añadirán manualmente. *Referencia [3]*.

2.3 Visual Basic Editor

Este será el entorno sobre el que se desarrollará el proyecto. Para tener acceso a éste, como ya se ha comentado, se entra en *Tools* → *Macro* → *Visual Basic Editor*, apareciendo la siguiente ventana:

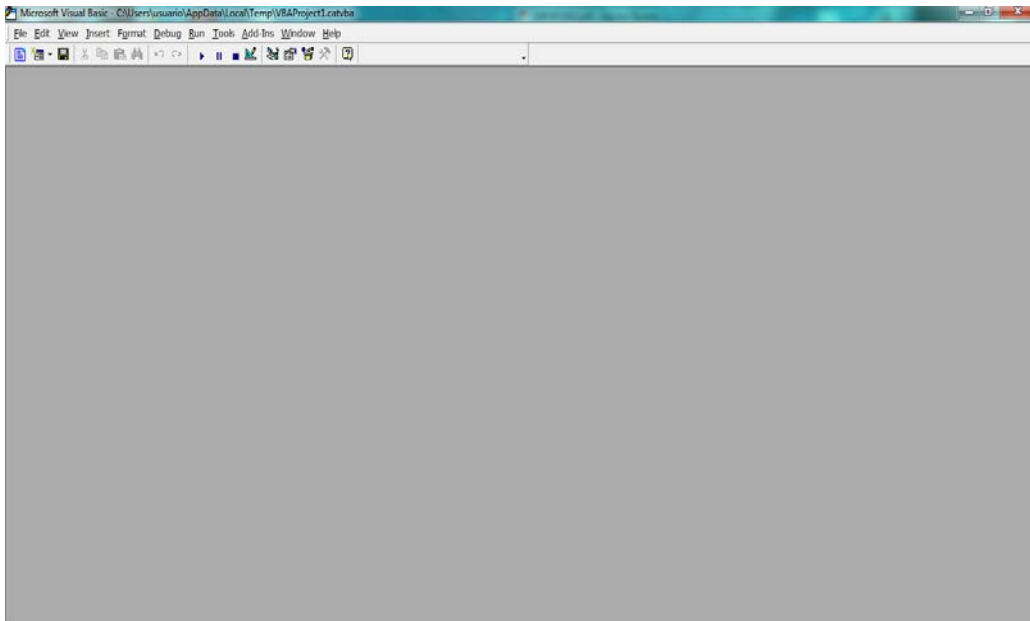


Figura 7. Entorno VBA.

Aquello que se programe y realice dentro de esta ventana puede interactuar directamente con CATIA si se usan los objetos del programa. Se mostrará seguidamente cada una de las partes o bloques que conforman el entorno donde se desarrolla toda la programación.

En primer lugar, es importante tener a vista del programador tanto el menú denominado *Project Explorer*, así como el *Properties Windows*, ya que ambos son menús fundamentales en los que se trabaja y los cuales facilitan mucho la tarea. Para acceder a ellos, se hace clic sobre la pestaña *View* y se adjuntan ambas aplicaciones a la pantalla principal tal como se muestra en la Figura 8.

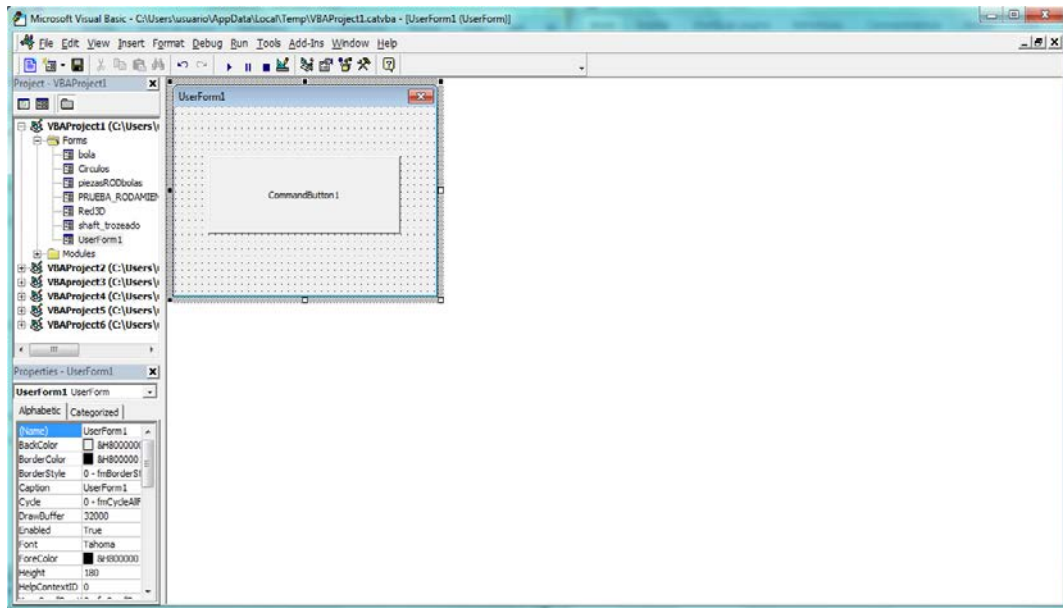


Figura 8. Project Explorer y Properties Window.

La ventana *Project Explorer* acumula todos los datos y archivos que componen el *Project* (proyecto de programación) en el que se esté trabajando. Puede contener formularios (*Userforms*), módulos (*Module*) y clases (*Class module*), que se explican a continuación:

- Módulos: son rutinas independientes. Dado que el código contenido en un módulo estándar de *Visual Basic* es abordable desde distintos formularios del programa, será útil colocar en este módulo todo lo que se quiera disponer como “código compartido”, es decir, que pueda ser utilizado en cualquier formulario del programa. Se guardan para su exportación en ficheros con extensión **.bas*.
- Formularios: son rutinas asociadas a ventanas gráficas donde se incorporan objetos y eventos. Los formularios son el elemento básico que permite la interacción del programa con el usuario, demandando variables, opciones, etc. Se almacenan con extensión **.frm*.
- Clases: son definiciones de nuevos objetos de tipo plantilla donde se definen las propiedades y eventos del mismo, son almacenados como **.cls*.

En lo que se refiere a *Properties Windows*, muestra en una columna todas las propiedades de cada formulario o control que se tenga seleccionado en la ventana *Project Explorer*. Dicha información se obtiene pinchando sobre el formulario o control deseado.

Referencia [2].

Para concluir, se va a explicar otra ventana que ha sido útil, la *Object Browser*, a la cual se llega de la misma forma que las anteriores.

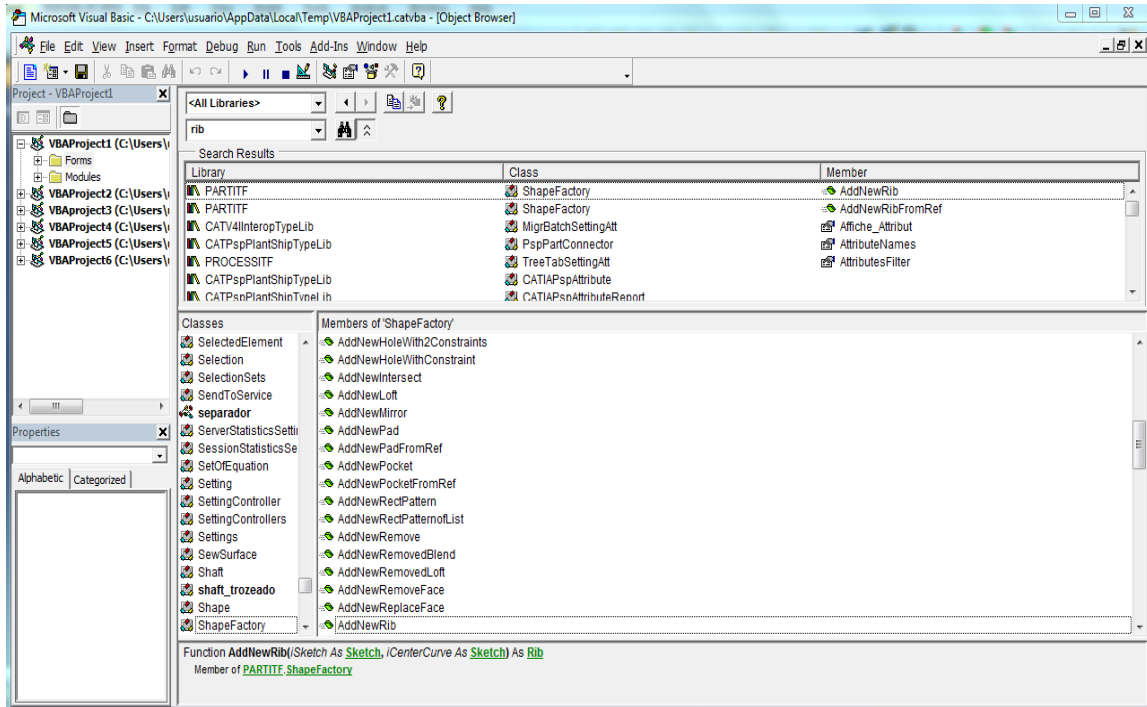


Figura 9. Object Browser.

Esta herramienta presta un gran servicio cuando se está en un punto de la programación en la que no se sabe bien qué hacer, cómo seguir o cómo funciona cualquier tipo de objeto. Si se escribe cualquier objeto en la barra de búsqueda, se muestran tres columnas que contribuyen a controlar dicho objeto. La tercera de ellas, la columna *Member*, expone distintas formas para realizar la misma operación, eligiéndose la más conveniente. La segunda columna, muestra a qué tipo de objeto pertenece la operación que se pretende realizar, por lo cual se sabrá qué objeto hay que definir para tener acceso a la herramienta deseada. Referencia [4].

Observando la parte inferior de la imagen anterior se muestra una descripción que sirve de ayuda ya que separa uno a uno todos los argumentos de entrada que son imprescindibles para realizar la operación. Para regresar a la ventana de trabajo no hay más que cerrar ésta.

2.4 Iniciación a la programación

Para iniciarse en este punto es primordial conocer que *Visual Basic* es un lenguaje de programación por eventos. La ejecución del programa se va desarrollando en diferentes secciones como respuestas a eventos que se producen o bien por la interacción con el usuario mediante una interfaz gráfica al que solicita los parámetros necesarios para la ejecución de una determinada acción o bien debido a la existencia de otras aplicaciones que desencadenan dichos eventos.

Por eso, a la hora de programar, habrá que prestar especial atención a cómo se está escribiendo el código para que responda adecuadamente a los eventos para los que se diseñe la aplicación.

Seguidamente se explican una serie de conceptos que es necesario conocer ya que en ellos se fundamenta la programación controlada por eventos.

Concepto	Definición
Tiempo de diseño	Instante en que se crea la aplicación.
Tiempo de ejecución	Instante en el cual se ejecuta y se interactúa con la aplicación.
Formulario	Ventana sobre la que es posible personalizar la interfaz de la aplicación o cuadro de diálogo para obtener información del usuario.
Objetos	Formularios y controles.
Controles	Representación gráfica de objetos, con lo que el usuario interactúa y aporta la información que se le pide.
Propiedades	Los valores de un objeto (<i>Properties Window</i> antes mencionada). Son características de un objeto y definen el estado del mismo en un momento específico.
Métodos	Las acciones que un objeto puede realizar sobre sí mismo. Se suelen usar verbos para dar nombre a los métodos.
Eventos	Acciones que son reconocidas por un formulario o control. Los eventos ocurren a medida que el usuario interactúa con los objetos de la aplicación.
Colección	Grupo o lista de objetos similares que se ponen juntos por una razón específica. Las colecciones son objetos que agregan un conjunto de otros objetos.
Clases	Definen un tipo de objeto. Se suele usar la herencia para crear jerarquía entre clases y subclases.
Herencia	Todas aquellas clases que sean herencia de la misma clase tienen todas las propiedades y métodos en

	común de la herencia de la que provienen, pero también tienen sus propios métodos y propiedades que las diferencian entre ellas.
--	--

Tabla 1. Conceptos de la programación.

Referencia [2] y [3].

Otros conceptos que conviene destacar antes de programar son los siguientes:

2.4.1 Declaración de estamentos

Antes de empezar a trabajar con una variable, constante o herramienta, es necesario nombrarla, obteniendo así una primera información sobre el estamento que se va a establecer después.

Si no se define el tipo de variable, VBA declarará la variable de tipo *variant*, la cual puede aceptar cualquier tipo de variable. Hay que tener en cuenta que en excepcionales ocasiones se tiene una buena razón para usar una variable de este tipo. En la mayoría de los casos, se definirá un tipo de variable, haciendo de esta forma que el código se ejecute más rápido y reduciendo errores, lo cual se debe a que:

1. CATIA ejecutará el tipo de variable que se especifique.
2. A la hora de revisar el código, siempre se sabrá de qué tipo es cada variable y la intención por la cual se creó.

El comando que se usará para definir el tipo de variable con el que se trabajará es *Dim*, como se muestra en el siguiente ejemplo:

<code>Dim documents1 As Documents</code>
--

En él, mediante la instrucción *Dim* se declara y asigna espacio a una variable llamada *documents1* del tipo *Documents*. *Referencia[3]*.

2.4.2 Estamentos

Son una instrucción completa que puede contener *keywords* (*And, If, For, While, Sub, Function...*), operadores (+, -, *, /,...), variables, constantes y expresiones. Un ejemplo de estamento sería:

<code>Set partDocument1 = CATIA.ActiveDocument</code>

Donde el comando *Set* se usa para asociar la variable definida con un objeto.
Referencia [2].

2.4.3 Estamentos ejecutables

Se trata de acciones iniciales como:

```
Select1.Search("name='Optimization.Minimunvalue',all")
```

Referencia [2].

2.4.4 Funciones y subfunciones

Son una sucesión de estados que constituyen la operación deseada. Esta operación viene especificada en una función.

```
Sub mySubwithParameter(myParameter)  
    MsgBox myParameter  
End Sub
```

La diferencia entre *function* y *sub* es que la primera te devuelve un valor la segunda no. *Referencia [2].*

2.4.5 Estructuras condicionales e iterativas

2.4.5.1 Condicionales

Dado que durante la programación, frecuentemente se llega a situaciones en las que conviene utilizar condicionales para comprobar, por ejemplo, que un valor se encuentra en un intervalo aceptable, es conveniente hacer mención de la ejecución:

If...Then...End If

La estructura es la siguiente:

```
If [Condición] Then  
    [Estamento]  
ElseIf [Condición] Then  
    [Estamento de ElseIf]  
Else  
    [Estamento de Else]
```

End If

2.4.5.2 Iterativas

Otra de las estructuras que se usa habitualmente en la programación son aquellas en las que se ejecuta una misma operación varias veces consecutivas con el propósito de hallar un valor justo o que haga funcionar un programa hasta que se cumpla cierta condición. Se puede hacer esto de distintas formas:

1. *For ... Next*: cuando se quiere iterar un número de cosas dado.

```
Contador
For [Contador] = [Inicio] To [End] {paso a paso}
  [Estamentos]
Next
```

2. *While ... Wend*: cuando se itera hasta que el contador cumple cierta condición.

```
Contador
While [{Contador} Condición]
  [Estamentos]
Wend
```

3. *Do ... Loop*: cuando se itera siempre que se cumpla cierta condición.

```
Do [{While\Until} Condición]
  [Estamentos]
  [Exit Do]
  [Estamentos]
Loop
Do [Estamentos]
  [Exit Do]
  [Estamentos]
Loop [{While\Until} Condición]
```

4. *For Each... Next*: cuando se quiere iterar sobre objetos en una determinada colección. *Referencia [2]*.

2.4.6 Objetos orientados a la programación

¿Qué son los objetos?. Una parte de memoria en la que está volcada cierta información y metodología que posibilita operar con dicha información almacenada. Necesita un cierto código especial para hacer que funcione, ya que dispone de pautas definidas por métodos específicos tales como:

- 1) Pasar información a través de parámetros.
- 2) Operaciones de cálculo que pueden:
 - a. Modificar cierta parte de los datos iniciales.
 - b. Configurar ciertas operaciones que no están por defecto.
 - c. Devolver valores de los datos iniciales.
 - d. Retornar resultados de los cálculos usando tanto la información introducida externamente como de la contenida en el objeto.

2.4.7 Cómo definir un objeto

A cada objeto se le asigna una clase (*classe*) que sirve como plantilla en la que se recoge cómo operan los objetos y los datos que albergan. Conviene destacar que una clase puede ser utilizada para hacer funcionar uno o más objetos. *Referencia [2]*.

Clase	Objeto
Describe la estructura del objeto.	Es el resultado de la clase.
Es una plantilla	Tiene una copia de toda variable no-estática pero no de las variables tipo clase (<i>static</i>).
Especifica la representación de la información, el comportamiento, la interrelación (vía variables, métodos y <i>parents</i> -estructura lógica)	

Tabla 2. Diferencias entre clase y objeto.

2.5 Interfaz con el usuario

En este punto se hace mención a cómo crear nuestra propia interfaz para interactuar con el programa y así llegar a la programación por eventos vista con anterioridad.

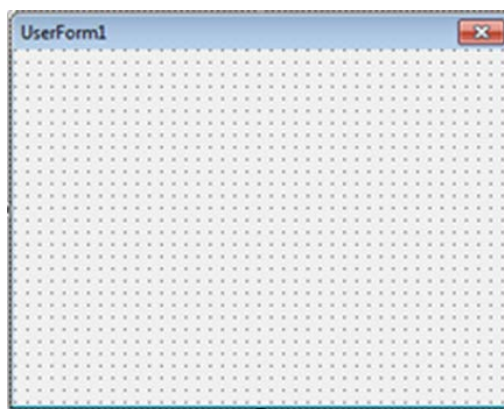


Figura 10. Formulario de trabajo.

Para ello se crea un nuevo formulario, mostrado en la Figura 10, sobre el que se diseñará la interfaz añadiendo los distintos controles indispensables para que el usuario interactúe con la aplicación. Para insertar un formulario abrimos la pestaña *Insert* → *Userform*.

Sólo resta añadir los elementos visuales que posibiliten al usuario controlar el programa. Para lo cual se activará la ventana “*Toolbox*”, que se hará desde la pestaña *View* → *Toolbox*.

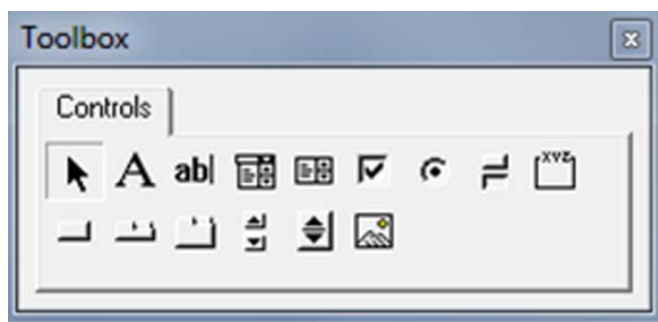


Figura 11. Herramientas de diseño.

Acto seguido, se explicarán los controles de los que se disponen para diseñar el formulario. *Referencia [2]*.

Símbolo	Etiqueta	Descripción
A	<i>Label</i>	Permite escribir títulos o comentarios.
abl	<i>TextBox</i>	Permite al usuario introducir texto.
☰	<i>ListBox</i>	Control en el que se muestran varios registros, pudiendo seleccionar uno o más de uno.












	<i>ComboBox</i>	Control parecido al ListBox con una propiedad llamada <i>Style</i> , que permite 3 formas distintas de presentar una lista.
	<i>CheckBox</i>	Permite seleccionar una opción al usuario.
	<i>OptionButton</i>	Permite seleccionar una opción al usuario.
	<i>ToggleButton</i>	Botón para selección de opciones.
	<i>Frame</i>	Agrupar diferentes objetos referidos a un mismo tema.
	<i>CommandButton</i>	Permite ejecutar un evento.
	<i>TabStrip</i>	Separadores o etiquetas.
	<i>MultiPage</i>	Contenedor para una colección de objetos.
	<i>ScrollBar</i>	Permite tener una barra para desplazamientos.
	<i>SpinButton</i>	Permite aumentar o disminuir la cifra conforme se presionan las flechas del control.
	<i>Image</i>	Insertar una imagen en el formulario.

Tabla 3. Opciones de la barra de herramientas.

Habrá que introducir y programar cada uno de ellos ya que la interfaz por sí misma no ejecuta ninguna operación. Para añadir dichos códigos sólo hay que hacer doble clic sobre el mismo formulario y se abrirá una ventana como la de la Figura 12, la cual se corresponde con un *CommandButton*.

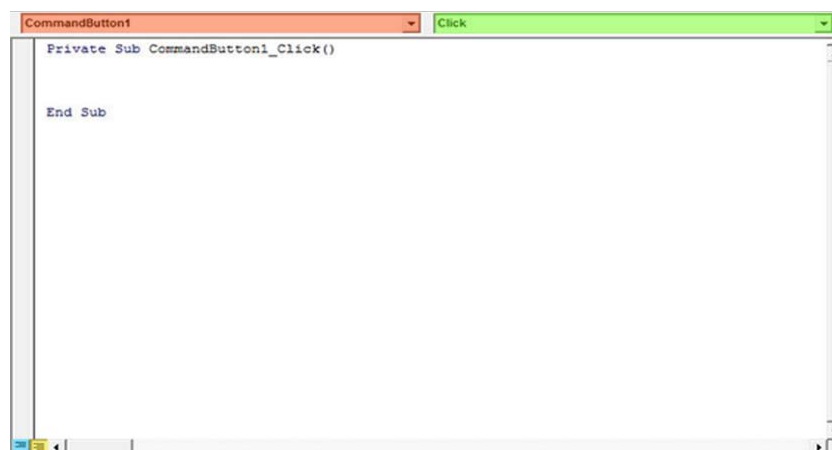


Figura 12. Editor.

Conviene describir la ventana de código adjunta. En ella se distinguen dos pestañas: una roja que recoge la lista de objetos disponibles para programar y una verde

que indica para el objeto seleccionado los eventos disponibles. Finalmente en la parte inferior se tienen los botones para ver el procedimiento y ver el módulo en azul y amarillo respectivamente.

En capítulos sucesivos se desarrollará cómo funcionan algunos de los controles mostrados en esta sección.

Para concluir, cuando se tiene la rutina diseñada, el siguiente paso será probarla. Para lo cual se dispone de una serie de opciones relativas al menú *Run*, también disponibles en el menú horizontal.



Figura 13. Comandos para hacer funcionar la aplicación.

Cuando se hace clic en el botón *Play*, el programa se traslada al entorno de CATIA y aparece en pantalla la aplicación creada.

3. Herramientas de programación de piezas.

3.1 Introducción

A continuación se van a presentar todos los procedimientos que se han llevado a cabo para la realización de las diferentes piezas que ayudarán a cumplir los objetivos.

Se trabajará con tres módulos, el módulo “*sketcher*”, para realizar los diferentes *sketches* necesarios, el módulo “*Part Design*”, para dar dimensión en 3D a los *sketches* y finalmente el módulo “*Assembly Design*” para unir las piezas o *parts* que se han creado con los módulos anteriores.

3.2 Arranque

Se comienza explicando cómo se define y establece el entorno de trabajo sin necesidad de tener abierto el *part*, lo cual se hará activando algunos parámetros que permitan al programador responder a las acciones que se le indican.

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

Código 1. Arranque de documentos.

El *PartDocument* activa el módulo *Part Design*, donde se definen el resto de elementos que componen el proyecto en el que se está trabajando. También se define el archivo *part* donde se creará la pieza, el cual se establece en el *PartDocument* definido anteriormente.

Una vez definido el *part*, es necesario crear el árbol de trabajo dónde se reflejarán todas las operaciones que se vayan realizando durante el proyecto, estableciendo el *body* de trabajo. Para ello, establecemos los *bodies*, que son como los hijos del *part*, por lo que primero habrá que definir el *bodies1* y dentro de estos uno específico *body1*.

Se puede observar la jerarquización de la estructura donde cada archivo está activado dentro de otro de manera que si alguno falla, otros no podrán realizar sus funciones ya que se produce el fallo global del programa.


```
Dim bodies1 As Bodies
Dim body1 As Body
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

Código 2. *Bodies*.

Al igual que hacemos cuando diseñamos manualmente cualquier pieza, lo primero que se va a realizar es un *sketch* donde poder dibujar. Dicho *sketch* se genera dentro del *body* definido anteriormente. A continuación hay que establecer el sistema de referencia 3D con el que se va a trabajar y además definir en qué plano de este sistema se va a hacer. Finalmente, colocamos el *sketch* en el que se dibujará dentro del plano de referencia definido anteriormente.

```
Dim sketches1 As Sketches
Set sketches1 = body1.Sketches
Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements
Dim reference1 As Reference
Set reference1 = originElements1.PlaneYZ
Dim sketch1 As Sketch
Set sketch1 = sketches1.Add(reference1)
```

Código 3. *Sketches*.

Para este código se ha cogido al azar el plano YZ y se ha guardado dentro de una referencia que hemos llamado *reference1*.

Con el *sketch* y el plano de trabajo creados, lo siguiente es establecer una matriz que defina las direcciones dentro de cada plano, de forma que el programa tenga claro en qué plano de trabajo se encuentra.

```
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0# 'Vector unidad x en el plano X
arrayOfVariantOfDouble1(1) = 0# 'Vector unidad y en el plano X
arrayOfVariantOfDouble1(2) = 0# 'Vector unidad z en el plano X
arrayOfVariantOfDouble1(3) = 0# 'Vector unidad x en el plano Y
arrayOfVariantOfDouble1(4) = 1# 'Vector unidad y en el plano Y
arrayOfVariantOfDouble1(5) = 0# 'Vector unidad z en el plano Y
arrayOfVariantOfDouble1(6) = 0# 'Vector unidad x en el plano Z
arrayOfVariantOfDouble1(7) = 0# 'Vector unidad y en el plano Z
arrayOfVariantOfDouble1(8) = 1# 'Vector unidad z en el plano Z
Set sketch1Variant = sketch1
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
'Establece el sistema de ejes absoluto del sketch en 3D
```

Código 4. Vector de coordenadas.

Finalmente se escribe esta línea de código informando que el *sketch1* es el lugar de trabajo a partir de ese punto y las operaciones que se realicen serán sobre el mismo.

```
part1.InWorkObject = sketch1
```

A continuación se definen y establecen el conjunto de herramientas para realizar el dibujo 2D y todos los elementos geométricos necesarios.

```
' Establecimiento del conjunto de herramientas
' 2D y asignarlo al sketch de trabajo
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

'Se establecen los elementos geométricos
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

'Se define el sistema de ejes dentro del sketch
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

'Establecimiento de las direcciones horizontal y vertical
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
line2D1.ReportName = 1
Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")
line2D2.ReportName = 2

'Se cierra el sketch y se establece como objeto de trabajo
sketch1.CloseEdition
part1.InWorkObject = sketch1
part1.Update
```

Código 5. Objeto *Factory2D*.

Las últimas líneas del Código 5, cierran el *sketch* en cuestión, lo establece como objeto de trabajo, es decir, sobre el que se realizarán las operaciones para el modelado de un sólido, y lo carga en el *part*. Referencia [2].

4. Sketcher

Dentro de este módulo se puede encontrar una serie de objetos y colecciones que resultarán de gran utilidad en este proyecto.

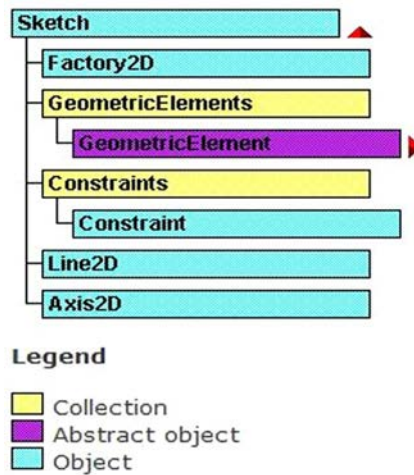


Figura 14. Estructura del módulo *Sketcher*.

Como se puede observar, los objetos que se encuentran en dicho módulo son “*Factory2D*”, “*Line2D*” y “*Axis2D*”, dentro de los cuales se encuentran una serie de métodos que se resumen a continuación:

Objeto	Comentario	Contenido
Factory2D	Incluye todos los métodos necesarios para poder trabajar en el módulo Sketcher.	CreateCircle
		CreateClosedCircle
		CreateClosedEllipse
		CreateControlPoint
		CreateEllipse
		CreateHyperbola
		CreateIntersection
		CreateIntersections
		CreateLine
		CreateLineFromVector
		CreateParabola
		CreatePoint
		CreateProjection
CreateProjections		
CreateSpline		
Line2D	Engloba tres métodos que dan el univector de la dirección de	GetDirection

	la recta, un punto sobre la línea y un último que permite modificar las características de la línea infinita.	GetOrigine
		SetData
Axis2D	Vienen detalladas las propiedades del sistema de coordenadas	HorizontalReference
		Origin
		VerticalReference

Tabla 4. Objetos del módulo *Sketcher*.

Una vez vista la estructura del módulo *sketcher*, se va a describir cómo usar los diferentes métodos que componen los principales objetos y colecciones de dicho módulo para diseñar la geometría de los diferentes perfiles en cuestión. *Referencia [2]*.

4.1 Crear un punto

Como se observa en la tabla 6, el objeto *factory2D* del módulo *sketcher* posee herramientas para crear una amplia gama de geometrías. Se va a describir a continuación cómo crear un punto. Para ello, se usará la siguiente línea de código:

```
Dim point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(coord_x, coord_y)
point2D1.ReportName = 3
point2D1.Construction = False
```

Código 6. Creación de un punto.

Como se puede apreciar, solo es necesario introducir las coordenadas del punto que se desea crear. A este punto se le puede asignar un nombre para que sea más fácil de localizar en el *sketch*, así como imponer que dicho punto no sea de construcción, que es como CATIA lo crea por defecto.

4.2 Crear una recta

Otra herramienta que se encuentra también en el objeto *factory2D* es la que permite crear una recta a partir de dos puntos, el de origen y el punto final, previamente creados. El código necesario para esto es el que sigue:

```
Dim line2D1 As Line2D
Set line2D1 = factory2D1.CreateLine(x1, y1, x2, y2)
line2D1.ReportName = 3
line2D1.StartPoint = point2D1
line2D1.EndPoint = point2D2
```

Código 7. Creación de una recta.

Los puntos de origen y final (*Point2D1*, *Point2D2*) se crean cómo se ha mostrado en el apartado anterior, y para que la recta quede bien definida, es necesario reseñar que dichos puntos sean el de origen y fin, de forma que el programa entienda que se trata de hacer una sola recta y no continúe con otra después.

4.3 Crear una elipse

Para concretar una elipse cerrada será necesario definir seis variables: la coordenada x e y del punto que será el centro de la elipse, las componentes x e y de la dirección del eje mayor, las cuales nos van a fijar la orientación de la misma, y la longitud de los ejes mayor y menor de la elipse.

Para ello es necesario crear primero el punto de origen, *point2D1*, con la orden *factory2D1.CreatePoint()* en las coordenadas que corresponda. Se usará la orden *factory2D1.CreateClasedEllipse()* para generar la elipse. Como se puede apreciar, el definir el punto sirve para que a través de *ellipse2D1.CenterPoint* el programa centre la circunferencia en dichas coordenadas. A continuación se muestran las líneas de código con las que se crea un círculo cerrado.

```
Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(x0, y0, x1, y1, 2a, 2b)
Ellipse2D1.CenterPoint = point2D1
ellipse2D1.ReportName = 4
```

Código 8. Creación de un círculo.

De igual manera, es muy útil saber crear arcos de elipse. Para ello se usa el comando *factory2D1.CreateEllipse(x0, y0, x1, y1, 2a, 2b, start, finish)*, donde las variables que hay que introducir son:

5. *x0*: coordenada horizontal del centro del arco de la elipse.
6. *y0*: coordenada vertical del centro del arco de la elipse.
7. *x1*: coordenada horizontal de la dirección del eje mayor del arco de la elipse.
8. *y1*: coordenada vertical de la dirección del eje menor del arco de la elipse.
9. *2a*: longitud del eje mayor del arco de la elipse.
10. *2b*: longitud del eje menor del arco de la elipse.
11. *start*: ángulo en radianes del punto donde comienza el arco.
12. *finish*: ángulo en radianes del punto donde finaliza el arco.

El siguiente código muestra la estructura para crear un arco de circunferencia.

```
Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateEllipse(x0, y0, x1, y1, 2a, 2b, ang_inicio
[radianes], ang_fin [radianes])
```

Código 9. Creación de un arco de circunferencia.

4.4 Restricciones

Para que el *sketch* quede bien definido y fijo, es decir, que sea inalterable por algún error cuando se esté creando o manipulando, es necesario establecer las restricciones o *constraints*.

Los *constraints* trabajan con referencias, es decir, hay que establecer la referencia de cada uno de los objetos del *sketch*. Una vez definidas éstas, es necesario una instrucción que permita relacionarlas entre sí, para lo cual se usa la siguiente línea de código: *constraintsX.AddBiEltCs(CatCsTypeDistance, reference1, reference2)*. En dicho comando se observan dos propiedades de gran interés.

La instrucción *AddBiEltCs* nos indica que la restricción va a usar dos referencias, es decir, se van a relacionar dos objetos. Se pueden relacionar uno, dos o tres objetos. Para ello solo se ha de cambiar el prefijo “Bi-” por el correspondiente. Es decir, una restricción tal que *AddMonoEltCs* solo necesitará una referencia y otra como *AddTriEltCs*, tres. En este proyecto se trabajará con restricciones de uno y dos objetos.

CatCsTypeDistance refleja el tipo de restricción que se quiere imponer. Existen numerosos tipos de restricciones como se muestra en la Tabla 5.

Con estas órdenes, el programa sabe que las referencias en cuestión se encuentran relacionadas según el tipo de restricción utilizada. Sin embargo no conoce la cantidad exacta del parámetro (el cual dependerá del tipo de restricción en cuestión) que las relaciona (distancia, ángulo...), por lo que hay que señalar que *constraint* tiene dos modos de trabajo:

1. Modo *constraint*: El valor asignado restringe la geometría en dicha posición
>>*constraint1.Mode = catCstModeDrivingDimension*
2. Modo *Measurement*: el valor solo refleja aquello que puede ser observado desde dicha posición

>> *constraint1.Mode = catCstModeDrivenDimension*

Número de referencias	Tipo
<i>BiEltCs</i>	<i>CatCstTypeAnnulContact</i>
	<i>CatCstTypeParallelims</i>
	<i>CatCstTypePerpendicularity</i>
	<i>CatCstTypeChamfer</i>
	<i>CatCstTypeConcentry</i>
	<i>CatCstTypeDistance</i>
	<i>CatCstTypeHorizontally/Vertically</i>
	<i>CatCstTypeLength</i>
	<i>CatCstTypeLineContact</i>
	<i>CatCstTypeMajor/MinorRadius</i>
	<i>CatCstTypeMidpoint</i>
	<i>CatCstTypeOn</i>
	<i>CatCstTypePlanarangle</i>
	<i>CatCstTypeDistance</i>
<i>CatCstTypeTangency</i>	
<i>MonoEltCs</i>	<i>CatCstTypeRadius</i>
<i>TriEltCs</i>	<i>CatCstTypeSimetry</i>

Tabla 5. Restricciones.

A continuación se van a explicar las restricciones más empleadas en el desarrollo de este proyecto. *Referencia [2]*.

4.4.1 **CatCstTypeDistance**

Esta restricción se usa para fijar la posición respecto al origen de coordenadas o la distancia entre dos objetos. También es posible utilizarlo para determinar la longitud de algunos objetos.

Además de los comandos introducidos al inicio de la subsección, es necesario declarar el valor de una longitud que será definida como la dimensión de la restricción.

```

'Establecimiento de las constraints
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
'constraint de la distancia horizontal de un punto al origen de coordenadas.
Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(point2D3)
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)
Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeDistance, reference7,
reference8)
constraint4.Mode = catCstModeDrivingDimension

```

```
Dim length2 As Length
Set length2 = constraint4.Dimension
length2.Value = x1
```

Código 10. *CatCstTypeDistance*.

En este código se describe la restricción que fija la distancia entre un punto, *reference7*, y una recta, *reference8*.

4.4.2 CatCstTypeRadius

La utilidad de esta restricción es poder fijar el radio de circunferencias y arcos de circunferencias. De nuevo habrá que definir la longitud y darle el valor del radio.

```
'establecimiento de los constraints
Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(circle2D1)
Dim constraint13 As Constraint
Set constraint13 = constraints1.AddMonoEltCst(catCstTypeRadius,
reference25)
constraint13.Mode = catCstModeDrivingDimension
Dim length8 As Length
Set length8 = constraint13.Dimension
length8.Value = r1
```

Código 11. *CatCstTypeRadius*.

Aquí se describe la restricción que fija el radio que tiene una determinada circunferencia, *circle2D1*.

4.4.3 CatCsTypeHorizontally/Vertically

Cuando queremos crear una recta que sea horizontal o vertical y no conocemos los puntos de inicio y final ni la longitud de esta, es muy útil definir una recta arbitraria e imponer esta restricción de horizontalidad o verticalidad, tal y como se hace en el Código 12 para definir la verticalidad.

```
Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D8)
Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D2)
Dim constraint12 As Constraint
Set constraint12 = constraints1.AddBiEltCst(catCstTypeVerticality,
reference22, reference23)
constraint12.Mode = catCstModeDrivingDimension
```

Código 12. *CatCstTypeHorizontally/Vertically*.

4.4.4 CatCstTypeLength

Por el contrario, si lo que queremos fijar no es la orientación de la recta sino la longitud de ésta, se puede emplear este *constraint*. Para ello habrá que crear una longitud y darle valor a esta.

```
Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D6)
Dim constraint10 As Constraint
Set constraint10 = constraints1.AddMonoEltCst(catCstTypeLength,
reference19)
constraint10.Mode = catCstModeDrivingDimension
Dim length7 As Length
Set length7 = constraint10.Dimension
length7.Value = d
```

Código 13. *CatCstTypeLength*.

En el presente código se fija la longitud de la recta definida como *line2D6* mediante *length7*.

4.4.5 CatCstTypeParallelism

Para fijar que una recta sea paralela a otra, se definen las dos rectas como referencias y se aplica la restricción de paralelismo a las dos referencias creadas.

```
Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D7)
Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D5)
Dim constraint11 As Constraint
Set constraint11 = constraints1.AddBiEltCst(catCstTypeParallelism,
reference20, reference21)
constraint11.Mode = catCstModeDrivingDimension
```

Código 14. *CatCstTypeParallelism*.

El Código 14 describe la imposición del paralelismo entre las rectas *line2D7* y *line 2D5*.

4.4.6 CatCstTypeOn

La restricción *CatCstTypeon* es una de las más interesantes de las restricciones existentes.

Dicha restricción permite imponer la coincidencia de dos puntos o curvas cualesquiera, por lo que permite introducir imposiciones geométricas muy importantes. En el código que se muestra a continuación, se está creando una coincidencia entre un punto y una recta. Como se puede apreciar, primero se hacen referencias a los ejes de las piezas y luego se realiza la restricción de coincidencia.

```
Dim reference7 As Reference
Set reference7 = product1.CreateReferenceFromObject(point2D5)

Dim reference8 As Reference
Set reference8 = product1.CreateReferenceFromName(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeOn,
reference7, reference8)
```

Código 15. *CatCstTypeOn*.

En este código se impone la restricción de coincidencia entre el punto *point2D5* y la recta *line2D1*.

5. CATPart

En esta sección de CATIA, denominada CATPart, vienen englobados tres módulos:

1. *PartDesign*.
2. *Wireframe and Surface Design*.
3. *Generative Shape Design*.

De estos módulos solo se usará en el presente proyecto el *PartDesign*, que contiene herramientas para diseñar cualquier tipo de geometría en formato sólido. A pesar de que no se empleen los demás, conviene tener una idea de dichos módulos para que el lector sepa dónde se encuentra situado dentro de CATIA. El *Wireframe and Surface Design* es un módulo que tiene órdenes de trabajo de superficies pero muy básico y el *Generative Shape Design* contiene las herramientas para conseguir cualquier tipo de forma basada en superficies conteniendo al anterior y órdenes más avanzadas. Se va a estudiar a continuación el primer módulo con profundidad.

5.1. Part Design

Este módulo se centra en otra sección de CATIA en el que se dispone de una gran variedad de objetos que permiten tener a mano toda la información que es necesaria para generar un modelo sólido.

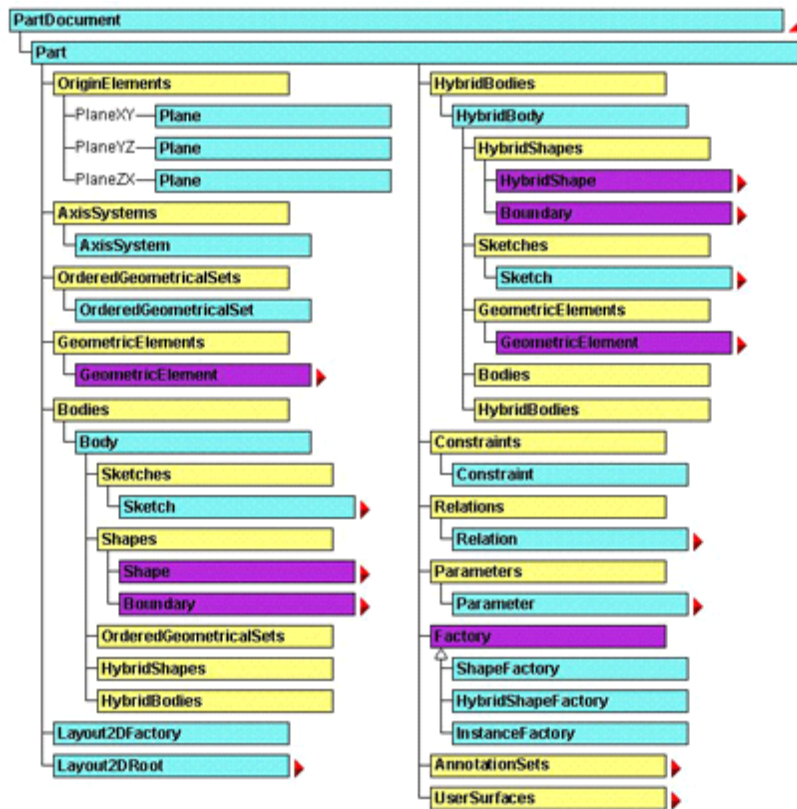


Figura 15. Estructura del módulo *Part Design*.

La estructura del *Part Design* es la que se aprecia en la Figura 15, donde los colores significan lo mismo que se ha explicado en el módulo previo.

Como ya se ha comentado anteriormente, la estructura de CATIA está muy jerarquizada, de manera que por ejemplo, para acceder a los planos de referencia, habrá que definir en primer lugar los objetos y propiedades referentes a *Plane* que contienen dichos planos (*PartDocument* → *Part* → *OriginElements*).



Figura 16. Estructura interna del *PartDocument*.

En la tabla siguiente se recogen las propiedades de los objetos que se utilizan en el módulo *Part Design*, explicando brevemente qué función tiene cada una de ellas.

Propiedad	Comando	Comentario
<i>OriginElements</i>	<code>Dim origin1 As OriginElements</code> <code>Set origin1 = part1.OriginElements</code>	Constituye el sistema de referencia absoluto del documento de planos XY, XZ e YZ.
<i>AxisSystems</i>	<code>Dim refsist1 As AxisSystems</code> <code>Set refsist1 = part1.AxisSystems</code>	Es la colección de sistemas de referencia que pueden existir en el documento <i>part</i> .
<i>GeometricElements</i>	<code>Dim geometric1 As GeometricElements</code> <code>Set geometric1 = Part1.GeometricElements</code>	Colección de elementos geométricos 3D del <i>partDocument</i> generadas directamente en 3D, es decir, sin la mediación del módulo <i>Sketcher</i>
<i>Bodies</i>	<code>Dim Bodies1 As Bodies</code> <code>Set Bodies1 = Part1.Bodies</code>	Colección de cuerpos sólidos dentro del <i>partDocument</i>
<i>HybridBodies</i>	<code>Dim HybBod1 As HybridBodies</code> <code>Set HybBod1 = Part1.HybridBodies</code>	Colección de <i>OpenBodies</i> (elementos de referencia)
<i>Constraints</i>	<code>Dim Constraints1 As Constraints</code> <code>Set Constraints1 = Part1.Constraints</code>	Colección de restricciones geométricas y dimensionales del <i>partDocument</i> .
<i>Relations</i>	<code>DimRelations1 AsRelations</code> <code>SetRelations 1 = Part1.Relations</code>	Colección de relaciones del <i>partDocument</i> .
<i>Parameters</i>	<code>DimParameters1 AsParameters</code> <code>SetParameters1 = Part1.Parameters</code>	Colección completa de todos los parámetros del <i>partDocument</i> .

Tabla 6. Objetos del *Part Design*.

Referencia [2].

Se va a explicar ahora el objeto *Shape Factory*, que contiene todas las operaciones que permiten pasar de un dibujo 2D a un sólido 3D, o que una vez obtenido éste, realizar transformaciones sobre el mismo. Este objeto presenta una amplia gama de operaciones que podemos encontrar en la ventana *Object Browser* pero solo se van a incluir las más usuales y significativas que se usarán durante el proyecto.

Objeto	Comentario	Contenido
ShapeFactory	Incluye todos los métodos necesarios para poder trabajar en el módulo <i>Part Design</i> .	<i>AddnewCircPattern</i>
		<i>AddnewHole</i>
		<i>AddnewPad</i>
		<i>AddnewPocket</i>
		<i>AddnewSlot</i>
		<i>AddnewRib</i>
		<i>AddnewEdgeFilletWithConstantRadius</i>
		<i>AddnewShaft</i>

Tabla 7. Herramientas del *ShapeFactory*.

Ahora se van a mostrar en mayor detalle estas herramientas que conforman el *ShapeFactory* y que se han empleado para realizar los distintos componentes. Lo primero que hay que hacer es definir y cargar el objeto en cuestión y una vez cargado ya se podrá hacer uso de las herramientas que incluye. *Reference [10]*.

```
Dim ShapeFactory1 As ShapeFactory
Set ShapeFactory1 = part1.ShapeFactory
```

Código 16. Definición del objeto *ShapeFactory*.

5.1.1 Pad

La herramienta *pad* es la comúnmente usada para hacer extrusiones a partir de un *sketch*. Es muy potente ya que con dicha herramienta y con alguna más que explicaremos posteriormente, se pueden definir infinidad de sólidos. Usaremos las siguientes líneas de código para hacer un *pad*:

```
Dim pad1 As Pad
Set pad1 = ShapeFactory1.AddNewPad(sketch1, profundidad)
pad1.Name = "nombre1" ' Darle un nombre al pad
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
Dim length1 As Length
Set length1 = limit1.Dimension
length1.Value = espesor
```

Código 17. *Pad*.

Los argumentos que nos pide la herramienta son el *sketch* que se desea extruir y la profundidad que se quiere fijar. Otra reseña de importancia es la posibilidad de llamar al *pad* con el nombre que elijamos para luego no confundirlo con otro posible *pad*.

Es necesario restringir la altura del *pad*. Para ello hay que definir tanto el límite inferior del que tiene que partir el *pad* como la longitud de la profundidad. Habrá que darle un valor a la dimensión de la longitud. *Reference* [2].

5.1.2 Pocket

La estructura del *pocket* es prácticamente igual a la del *pad* pero en este caso, lo que se quiere hacer es un hueco sobre un sólido ya creado, de manera que, tendremos que hacer una referencia a la superficie sobre la que se quiere hacer dicho hueco.

La referencia se puede hacer de dos formas:

1. Creando una referencia directa de la superficie del sólido usando las líneas mostradas en el Código 18. Se observa que el *pocket* se hará sobre una superficie *RSur:Face*, especificando que se realiza sobre el *pad* ya creado.

```
Dim reference10 As Reference
Set reference10 =
part1.CreateReferenceFromName("Selection_RSUR:(Face:(Brp:(Pad.1;2);None:());~
~ Cf11:());Pad.1_ResultOUT;Z0;G3055)")
```

Código 18. Referencia del *pad* 1.

2. O bien creando un plano con un offset que coincida con la distancia a la superficie del sólido sobre la que se quiere generar el *pocket*. Para ello se crea un plano, herramienta que se encuentra dentro del objeto *HybridShapeFactory*, en el cual se encuentran las principales herramientas del módulo *Wireframe and Surface Design*. Esta forma será la empleada en el proyecto que aquí se expone.

```
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
Dim reference10 As Reference
Set reference10 =
part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Set hybridShapePlaneOffset1 =
hybridShapeFactory1.AddNewOffset(reference10, altura, False)
```

Código 19. Referencia del *pad* 2.

El procedimiento seguido consiste en crear un plano XY, que servirá de referencia a nuestro plano, indicando que son paralelos.

Cuando ya tenemos la referencia para hacer el *pocket*, el resto del proceso se realiza de manera análoga al caso del *pad*, usando el primero de los casos explicados anteriormente:

```
Dim reference10 As Reference
Set reference10 =
part1.CreateReferenceFromName("Selection_RSUR:(Face:(Brp:(Pad.1;2);None:());~
~ Cf11:());Pad.1_ResultOUT;Z0;G3055)")

Dim pocket1 As Pocket
Set pocket1 = ShapeFactory1.AddNewPocket(sketch2, espesor)
pocket1.Name = "hueco interno" Dar nombre al agujero

Dim limit2 As Limit
Set limit2 = pocket1.FirstLimit
limit2.LimitMode = catUpToSurfaceLimit ' Se define su profundidad a la
' superficie más próxima
```

Código 20. *Pocket*.

Reference [2].

5.1.3. Rectangular Pattern

Esta herramienta es una de las más difíciles de programar debido a la cantidad de argumentos de entrada que pide dicha función.

Dicha herramienta resulta de gran utilidad ya que permite hacer copias de un sólido o de otra herramienta como un *pad* o un *pocket*, en las direcciones del plano en el que se encuentra sin necesidad de crear un origen de coordenadas cartesiano.

La principal diferencia con respecto a las anteriores herramientas que se han usado para generar sólidos es que no pide un *sketch* como argumento de entrada de la función, sino que se necesita como partida un sólido, el cual se puede realizar con cualquiera de las operaciones descritas.

Se muestra primero el código para programar un *rectangular pattern*, y a continuación se analizarán todos los argumentos que necesitamos como entrada, así como los diferentes parámetros que se deben definir para el correcto funcionamiento de dicha herramienta.


```
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromName(hybridShapePlaneOffset1)
```

```
Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromName(hybridShapePlaneOffset1)
```

```
Dim rectPattern1 As RectPattern
Set rectPattern1 = ShapeFactory1.AddNewRectPattern(Forma, Copias en
dirección 1, Copias en dirección 2, Espaciado en dirección 1,
Espaciado en dirección 2, Número de copias en la dirección 1, Número de copias en la
dirección 2, referencia en la dirección 1, referencia en la dirección 2, orientación a
copiar en la dirección 1, orientación a copiar en la dirección 2, ángulo de rotación)
```

```
rectPattern1.FirstRectangularPatternParameters = catInstancesandSpacing
rectPattern1.SecondRectangularPatternParameters = catInstancesandSpacing
```

```
Dim linearRepartition1 As LinearRepartition
Set linearRepartition1 = rectPattern1.FirstDirectionRepartition
```

```
Dim length6 As Length
Set length6 = linearRepartition1.Spacing
```

```
Length6.Value = 100
```

```
Dim linearRepartition2 As LinearRepartition
Set linearRepartition2 = rectPattern1.FirstDirectionRepartition
```

```
Dim intParam1 As IntParam
Set intParam1 = linearRepartition2.InstancesCount
```

```
intParam1.Value = 25
```

```
Dim linearRepartition3 As LinearRepartition
Set linearRepartition3 = rectPattern1.SecondDirectionRepartition
```

```
Dim length7 As Length
Set length7 = linearRepartition3.Spacing
```

```
Dim linearRepartition4 As LinearRepartition
Set linearRepartition4 = rectPattern1.SecondDirectionRepartition
```

```
Dim intParam2 As IntParam
Set intParam2 = linearRepartition4.InstancesCount
```

```
intParam2.Value = 5
```

```
part1.UpdateObject rectPattern1
```

Código 21. *Rectangular Pattern.*

De las distintas opciones que permite escoger dicha herramienta, se ha explicado la que se ha usado en la realización del proyecto, la cual crea las copias que se desean basándose en los siguientes parámetros:

- Número de copias que se desean realizar.
- Espacio entre instancias.
- Dirección a la que se van a realizar las copias.

6. CATProduct

Una vez que se han creado todos los componentes de la figura final, hay que ensamblarlas para que en conjunto formen el producto que se desea y poder así analizar las posibles interferencias que haya entre las distintas piezas. Para esta tarea es necesario emplear la sección de CATIA llamada *CatProduct*, y dentro de ésta se encuentra un módulo muy potente que nos permitirá unir y ensamblar las piezas de nuestra sección central. Este módulo es conocido como *Assembly Design*.

6.1 Assembly Design

Las piezas que se van creando tienen que ser guardadas en una carpeta común para que cuando se vayan a unir, el módulo haga una llamada a las piezas y éstas se alojen en el árbol de trabajo.

El módulo *Assembly Design* dispone de una serie de objetos que resultan fundamentales para conseguir el objetivo de este proyecto. La estructura que presenta el *ProductDocument* es la que se muestra en la Figura 17.

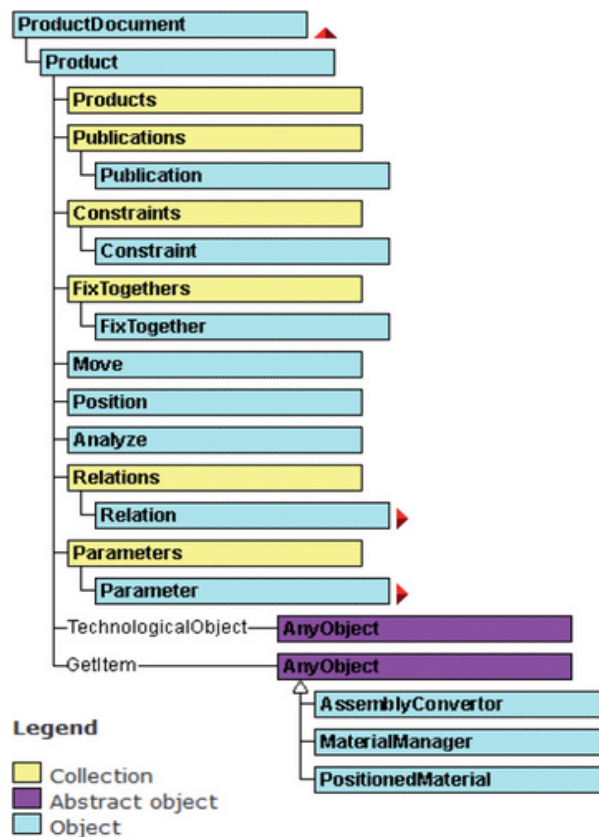


Figura 17. Estructura interna del *ProductDocument*.

Los parámetros, las fórmulas, las restricciones y publicaciones de los *CatProducts* están almacenados en los objetos llamados *Parameters*, *Relations*, *Constraints*, *FixTogethers*, y *Publications*.

Mediante este módulo, se van a imponer las restricciones necesarias entre los distintos *CatParts* creados para que el conjunto sea reconocido como un conjunto. A continuación se muestran las herramientas usadas en este proyecto. *Referencia [4]*.



Nombre	Icono	Función
<i>AddComponentsFromFiles</i>		Cargar <i>CatParts</i> , creados previamente, en el árbol de trabajo del <i>CatProduct</i> .
<i>Fix Component</i>		Fijar una pieza en la posición en la que se encuentra.

Tabla 8. Herramientas del módulo *Assembly Design*.

A continuación se muestran en más detalle las herramientas usadas así como unos códigos de cada restricción que sirvan de ejemplos aclarativos.

6.1.1 AddComponentsFromFiles

Lo primero que hay que hacer es seleccionar y añadir las piezas que se desean al *ProductDocument*. Todas estas piezas han sido guardadas en una carpeta, de la cual conocemos su directorio.

Para añadir dichas piezas al producto, solo habrá que decirle al programa que las busque en el directorio en el que se han guardado. En el código que se muestra, hemos guardado el aro externo de un rodamiento de contacto angular en una carpeta cuyo directorio se ha guardado con el nombre *objPath*.

Ejemplo: añadir piezas a un product.

```
Dim arrayOfVariantOfBSTR1(0)
arrayOfVariantOfBSTR1(0) = objPath & "\SecciónCentral.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR1, "All"
```

Código 22. *AddComponentsFromFiles*.

6.1.2 Fix Component

Es normal que cuando se monta un producto, se tome una pieza como referencia sobre la que se montarán el resto. Esta herramienta es la que permite fijar una pieza para tomarla como base sobre la que se ensambla el producto. En el presente trabajo se tomará

como referencia a la sección central del fuselaje.

Como ya hemos comentado, esta herramienta añade una restricción a la pieza. Esta *constraint* es la que mencionamos anteriormente en el capítulo 4: *catCstTypeReference*. Para usar dicha herramienta, solo hay que crear una referencia de la pieza y darla como argumento.

```
Dim reference1 As Reference
Set reference1 =
product1.CreateReferenceFromName("Product1/Part1.1/!Product1/Part1.1/")

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeReference,
reference1)
```

Código 23. *Fix Component*.

7. Aplicación: entorno de programación.

En esta sección se van a desarrollar el procedimiento que nos va a permitir analizar los posibles contactos y conflictos entre las piezas que van a formar nuestro producto final.

En primero lugar se va a crear un formulario, en el cual se va a poder elegir entre los distintos elementos que van a constituir el *product*. Entre ellos se encuentran la sección central del fuselaje, los asientos de los pasajeros, los estantes en los que guardar el equipaje de mano de cada uno de los mismos, carga en la bodega, baño y galley o catering. Dentro de cada uno de los elementos se exigirán una serie de datos fundamentales para el funcionamiento correcto de la aplicación. Es de suma importancia dar a conocer que toda la aplicación se ha desarrollado con la sección central como elemento principal, de modo que la definición de la misma será indispensable pues existen ciertos datos de partida procedentes de ella que se usarán para la caracterización del resto.

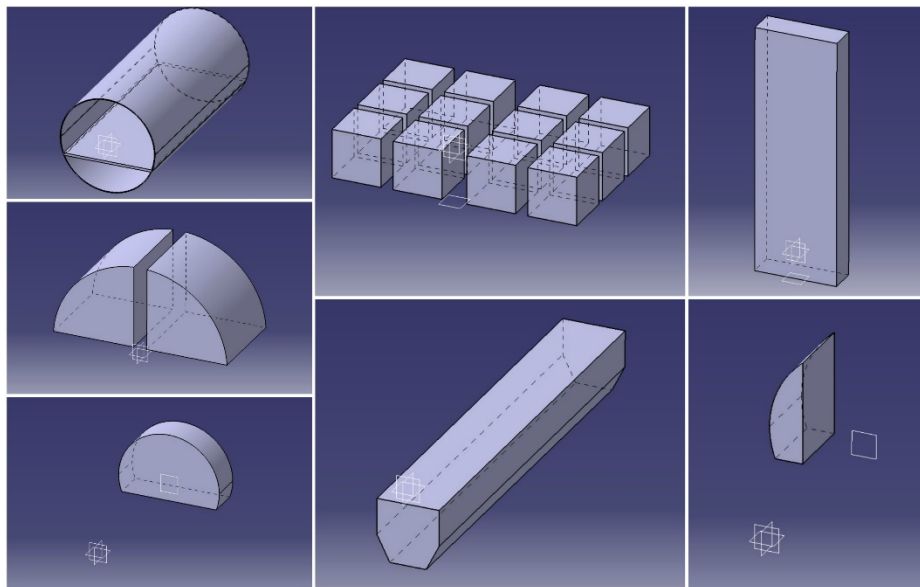


Figura 18. Elementos que van a constituir el *product*.

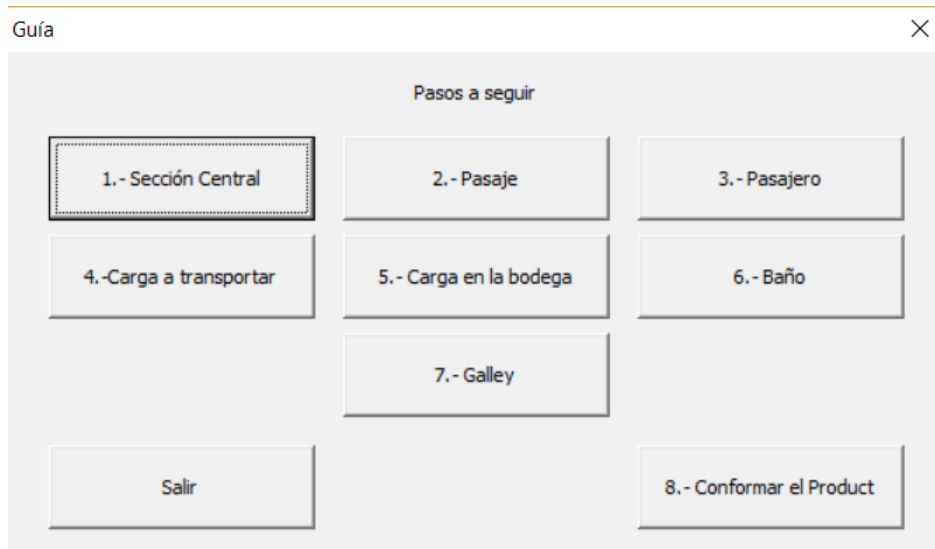


Figura 19. Interfaz del programa.

En el formulario principal se distinguen un solo tipo de control: *CommandButton* para acceder a cada uno de los elementos que se van a crear.

Cada uno de los *CommandButton* de este formulario está programado para que cuando se haga clic sobre él, el programa abra automáticamente otro formulario en el que o bien se deban introducir una serie de datos o bien se tenga que elegir entre opciones de forma que se acabe desembocando en el objeto el cual el usuario desea generar, por ejemplo, haciendo clic sobre el *CommandButton* correspondiente a la Sección Central aparecerá el formulario de la Figura 21. Este entramado de formularios que se explicarán más en detalle posteriormente hacen a la aplicación más versátil y provocan, al menos en su objetivo, que la experiencia del usuario sea satisfactoria.

Para interactuar entre los formularios, se han usado las siguientes líneas de código:

```
Private Sub CommandButton1_Click()
Load NombreUserform
NombreUserform.Show
End Sub
```

Código 24. Llamada a los *Userforms* con un *CommandButton*.

Habrá un *CommandButton* y por lo tanto un formulario asociado a cada elemento:

- Sección central.
- Pasaje.
- Pasajero.

- Carga a transportar.
- Carga en la bodega.
- Baño.
- Galley.

Una vez definida la interfaz de trabajo, se van a explicar los procedimientos que se han llevado a cabo para que dicho programa funcione correctamente.

7.1 Ventana emergente para seleccionar la carpeta de guardado

Como ya se ha mencionado anteriormente, las piezas que se van creando tienen que ir siendo almacenadas en una carpeta para que cuando sean llamadas por el *Product*, se puedan añadir al trabajo. Es obligatorio que todos los *Parts* sean guardados en la misma carpeta ya que de otra forma, el *Product* no funciona correctamente.

Los códigos que permiten desplegar una ventana emergente son los siguientes:

```
'=====
'codigo para seleccionar la carpeta donde guardar los archivos
'=====
Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath

Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE,
strTitle, NO_OPTIONS)

Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing
```

Código 25. Ventana emergente I.

Y la ventana emergente tiene la siguiente forma:

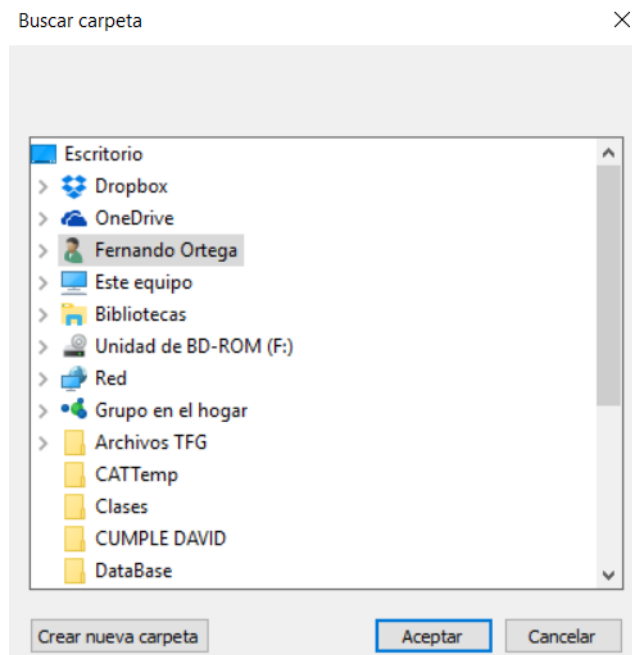


Figura 20. Guardado I.

En este código podemos ver que la herramienta *BrowseFolderDialogBox* es la que despliega la ventana y que cuando seleccionamos cualquier carpeta, el directorio de dicha carpeta o *Folder*, se almacena en lo que en el código se llama *objPath*. También tenemos la opción de crear una nueva carpeta en dicha ventana.

Cuando ya se tiene la carpeta donde se irán almacenando las piezas, solo falta ir guardándolas. Para ello se usa la línea de código siguiente.

```
partDocument1.SaveAs objPath & "\\NombreDelPart.CATPart"
```

Código 26. Guardado I.

Conviene comentar también que para las piezas que se generaban a través de un bucle, como se verá más adelante, se ha decidido emplear otra forma más cómoda y cuyo uso no genera errores que puedan provocar el mal funcionamiento de la herramienta como sí ocurre con la ventana emergente anterior. El bloque de instrucciones se muestra a continuación así como una imagen de la ventana emergente.

```
'=====
'guardado del Part
'=====

Dim RutaPart As String
Dim MensajePrecaución As String
```

```
RutaPart = CATIA.FileSelectionBox("SaveAs", "*.CATPart", 1)
```

'Posible Cancelación

```
If RutaPart = "" Then
```

```
MensajePrecaución = "La operación de guardado fue cancelada"
```

```
MsgBox MensajePrecaución, 48, "Part no guardado"
```

```
Exit Sub
```

```
End If
```

'Guardado

```
documents1.SaveAs RutaPart
```

Código 27. Ventana emergente II.

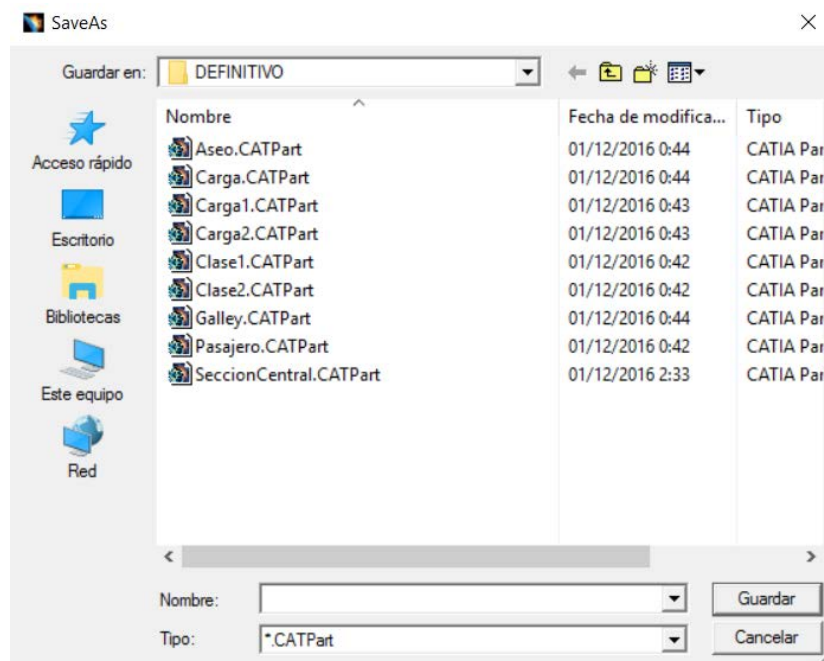


Figura 21. Guardado II.

7.2 Sección central

Al acceder desde el formulario principal al correspondiente a la sección central nos encontramos con la siguiente interfaz:

Características de la sección central ×

Radio Mayor	<input type="text"/>	mm
Radio Menor	<input type="text"/>	mm
Espesor	<input type="text"/>	mm
Altura del suelo	<input type="text"/>	mm
Espesor del suelo	<input type="text"/>	mm
Profundidad	<input type="text"/>	mm

Figura 22. Formulario de la sección central.

En ella nos aparecen los datos geométricos que van a definir nuestra sección central, los cuales se introducirán en mm:

- Radio Mayor: identifica el valor introducido en el *TextBox* con el radio mayor de la elipse que se pretende generar.
- Radio Menor: identifica el valor introducido en el *TextBox* con el radio menor de la elipse que se desea generar.
- Espesor: identifica el valor introducido en el *TextBox* con el espesor de la estructura de la sección central.
- Altura del suelo: identifica el valor introducido en el *TextBox* con la altura a la que se quiere colocar el suelo, tomando como referencia el punto exterior más bajo de la sección.
- Espesor del suelo: identifica el valor introducido en el *TextBox* con el espesor del suelo que dividiría la zona habitable de la bodega de carga.
- Profundidad: identifica el valor introducido en el *TextBox* con la extensión a la que se va a extruir toda la estructura.

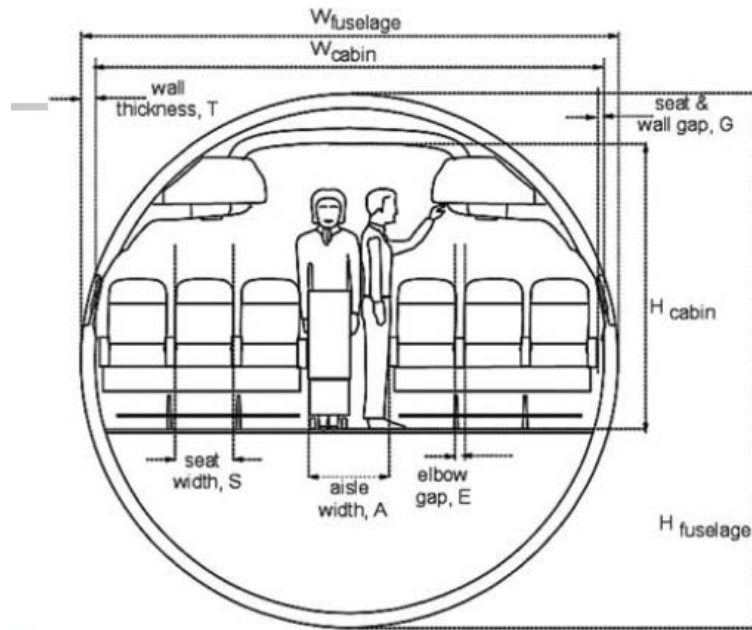


Figura 23. Características de la sección central.

La interfaz asociada a la sección central es la que se muestra en la Figura 21.

El desarrollo de la pieza que se está describiendo es el siguiente:

Empezar con un *sketch* de dos elipses concéntricas a las que, mediante el objeto *ShapeFactory*, se le hace un extrusionado con la herramienta *pad* en sentido inverso al que propone CATIA predeterminadamente, de una determinada profundidad. Basta con hacer iguales tanto el radio mayor de la elipse como el menor para que la forma resultante sea una circunferencia. Al espacio que existe entre ambas elipses se le ha denominado espesor.

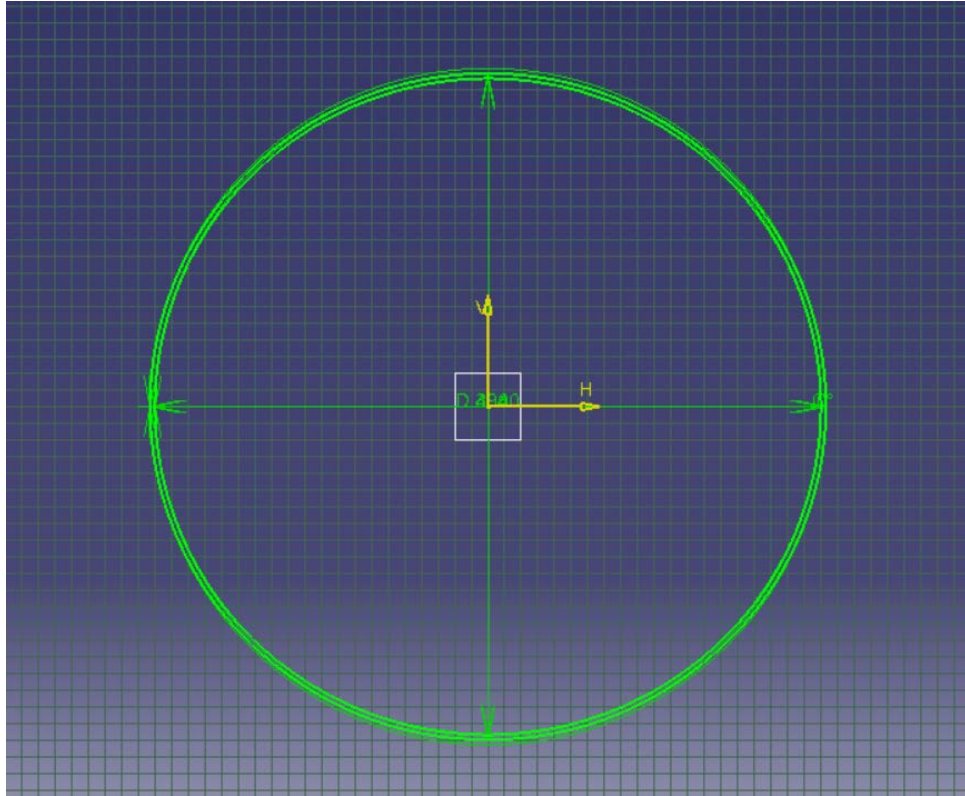


Figura 24. Creación de las elipses concéntricas.

Una vez realizado esto, se dibuja otro *sketch* sobre la cara visible de la estructura. En dicha cara, se dibuja un rectángulo de altura el espesor del suelo que se quiere disponer ajustando las esquinas inferiores del mismo a la elipse exterior de la estructura tal y como se observa en la Figura 25. Luego mediante el objeto *ShapeFactory* se le hace un extrusionado a través de la herramienta *pad* tal y como se hizo anteriormente obteniéndose así la composición deseada.

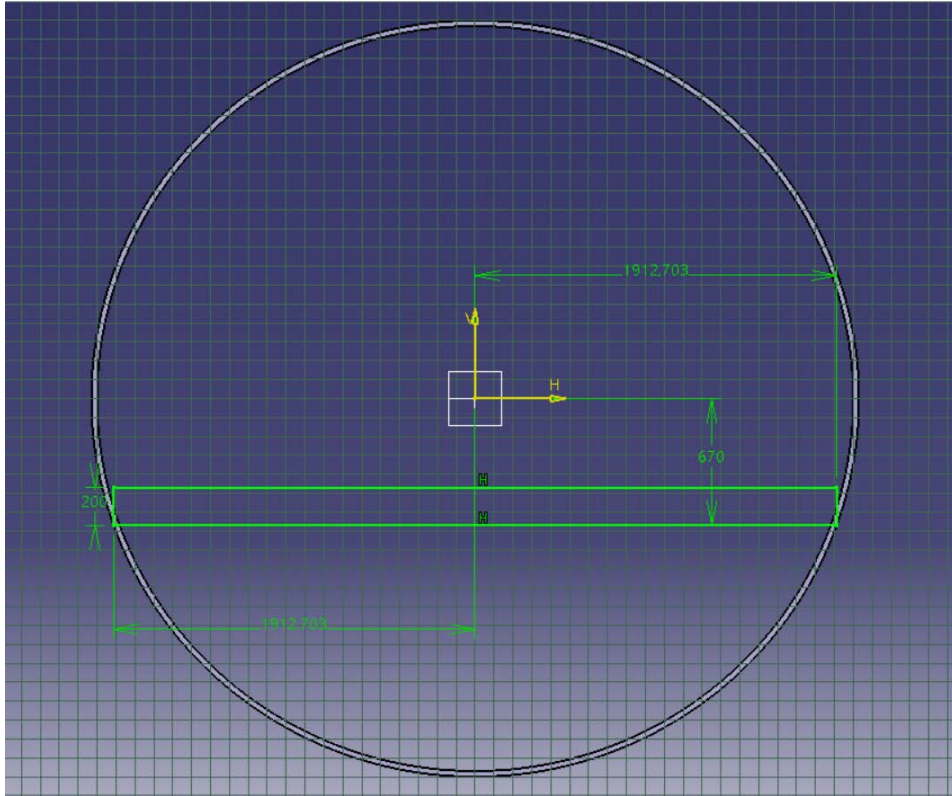


Figura 25. Suelo de la cabina.

Por último se abrirá una ventana en la que se podrá proceder al guardado del *part*. Se considerará conveniente guardar todos los *parts* en la misma carpeta tal y como se expuso antes. A partir de ahora no se explicitará el guardado de los *parts* en la misma carpeta para que el texto no resulte tan redundante.

El resultado final es el siguiente:

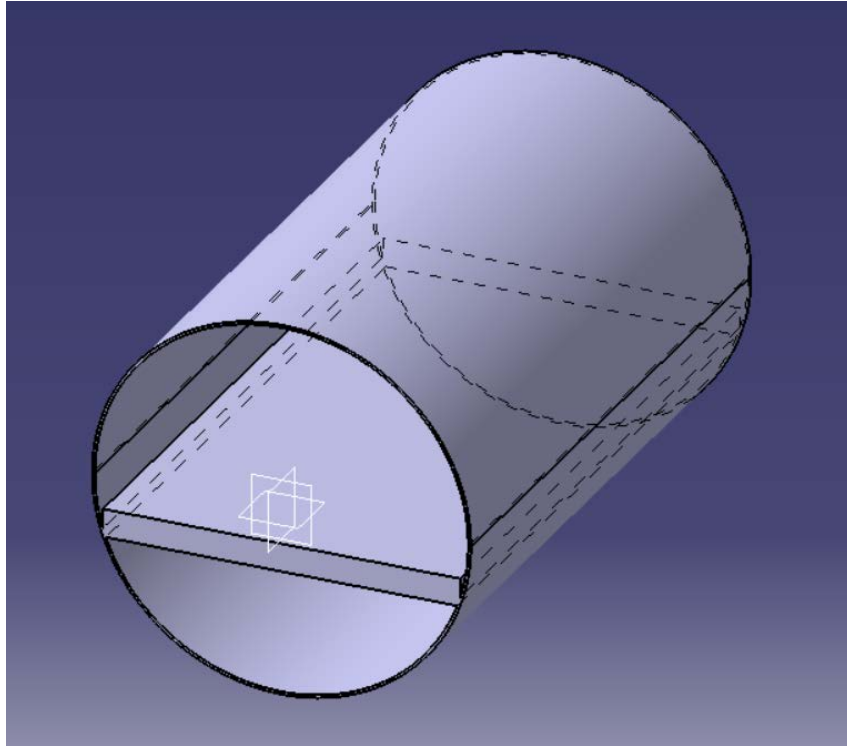


Figura 26. Sección central del fuselaje.

7.3 Pasaje

Al hacer clic sobre el *CommandButton* correspondiente al pasaje aparecerá el siguiente formulario:

Un formulario de software con el título "Elección del número de clases" y un botón de cerrar "X" en la esquina superior derecha. El formulario contiene un campo de entrada de texto etiquetado "Número de clases" con un cursor de texto visible. Debajo del campo de entrada hay dos botones: "Importar datos" a la izquierda y "Atrás" a la derecha.

Figura 27. Formulario correspondiente a la elección del número de clases.

En este formulario tan simple sólo hay que introducir el número de clases de las que queremos que disponga nuestro habitáculo para poder distinguir los asientos que pertenezcan a dichas clases tal y como se hace en la aviación comercial en la actualidad. La forma de los asientos se ha simplificado a prismas para que la generación sea más

sencilla y porque su nivel de definición no es relevante en este trabajo. Tras haber importado los datos, se desencadenará otro formulario que se muestra en la siguiente figura un número de veces igual al número de clases que se haya introducido.

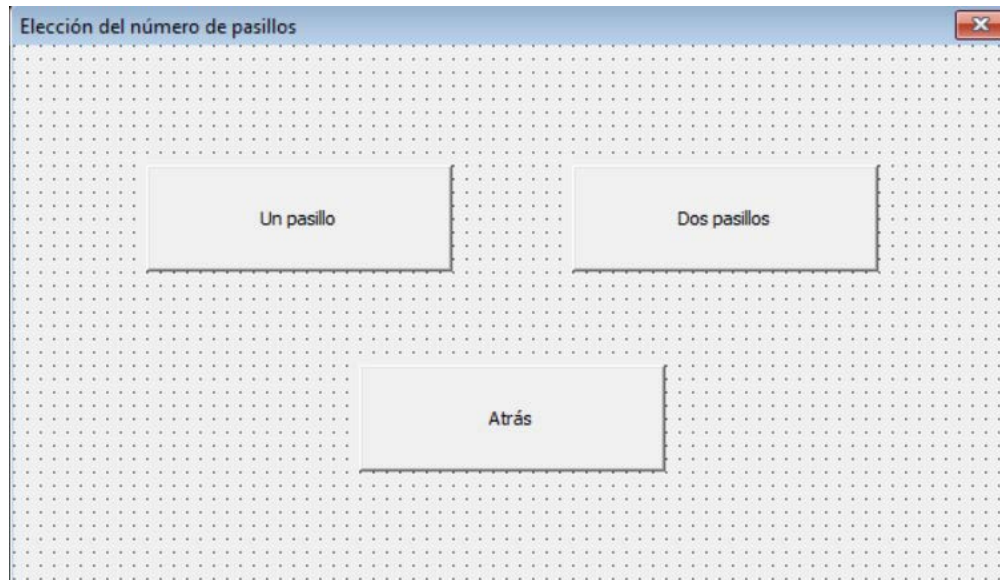


Figura 28. Formulario correspondiente a la elección del número de pasillos.

La acción de ejecutar un número determinado de veces el formulario se hace a través del Código 28. La interfaz que se presenta en la Figura 28 da la opción de elegir el número de pasillos con el que se quiera caracterizar a la clase que corresponda, eligiendo entre uno o dos pasillos. Conviene hacer la distinción de los parámetros que ofrece cada una de las opciones, hecho que se hará en los siguientes apartados.

```
Dim Nc As Double  
Nc = TextBox1.Value  
For i= 1 To Nc  
EligePasillo.Show  
Next i  
EligeClases.Hide  
VentanaPrincipal.Show
```

Código 28. Bucle empleado para la caracterización de las clases.

7.3.1 Pasaje de un pasillo

La disposición del pasaje con un pasillo, como su nombre indica, consiste en distribuir los asientos a un lado y a otro del mismo. El formulario correspondiente, el cual se expone a continuación, presenta todas las variables necesarias para caracterizar por completo la clase objeto de definición.

Pasaje de un pasillo ×

Offset delantero	<input type="text"/>	mm
Número de filas	<input type="text"/>	
Número de asientos por fila	<input type="text"/>	
Altura del asiento	<input type="text"/>	mm
Anchura del asiento	<input type="text"/>	mm
Longitud del asiento	<input type="text"/>	mm
Pitch	<input type="text"/>	mm
Separación entre asientos	<input type="text"/>	mm
Anchura del pasillo	<input type="text"/>	mm

Figura 29. Formulario correspondiente al pasaje de un pasillo.

El desglose de los datos que se necesitan introducir es el siguiente:

- Offset delantero: identifica el valor introducido en el *TextBox* con la distancia desde el plano YZ al lugar en el cual se van a empezar a construir los asientos. La siguiente ilustración sirve para clarificar la definición.

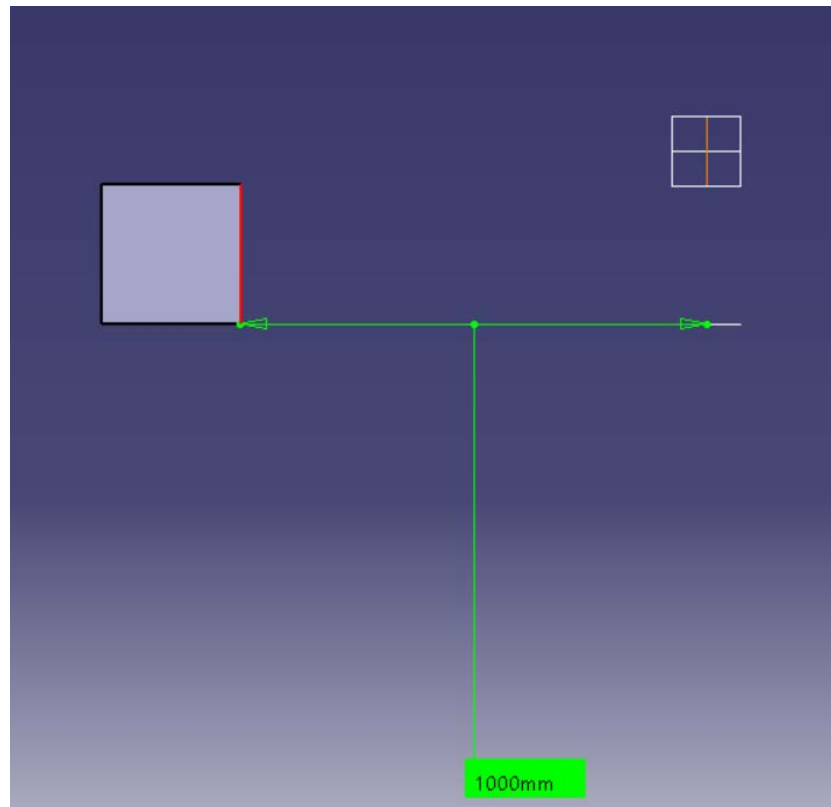


Figura 30. Offset delantero.

- Número de filas: identifica el valor introducido en el *TextBox* con el número de filas de asientos que se quiere disponer a la clase que se está caracterizando.
- Número de asientos por fila: identifica el valor introducido en el *TextBox* con número de asientos por fila que se quiere disponer a un lado y al otro del pasillo a la clase que se está caracterizando.
- Anchura del asiento: identifica el valor introducido en el *TextBox* con la dimensión del asiento considerada de derecha a izquierda o de izquierda a derecha, en contraposición a la considerada de arriba abajo o de abajo arriba en una superficie.
- Altura del asiento: identifica el valor introducido en el *TextBox* con la distancia vertical del asiento a la superficie tomada como referencia, en este caso el suelo de la cabina.
- Longitud del asiento: identifica el valor introducido en el *TextBox* con la extensión del asiento sobre la superficie tomada como referencia, en este caso el suelo de la cabina.

- Pitch: identifica el valor introducido en el *TextBox* con la distancia entre el respaldo de un asiento y el mismo punto del respaldo posterior. En la Figura 31 se muestra la definición de pitch.

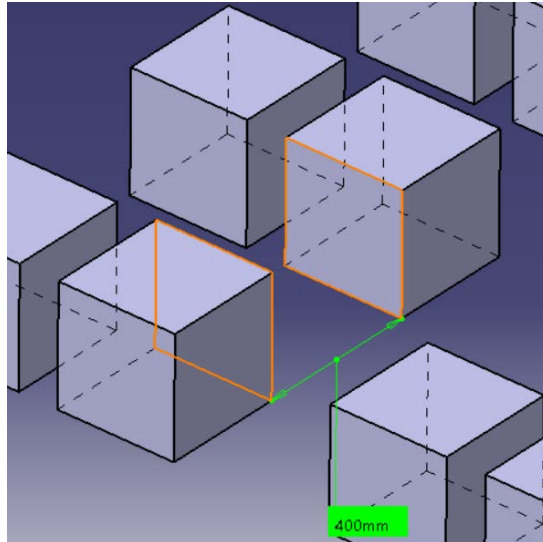


Figura 31. Pitch.

- Separación entre asientos: identifica el valor introducido en el *TextBox* con la distancia considerada de izquierda a derecha o derecha a izquierda que hay entre los asientos de la misma fila a un lado o al otro del pasillo. Esto se expone en la Figura 32.

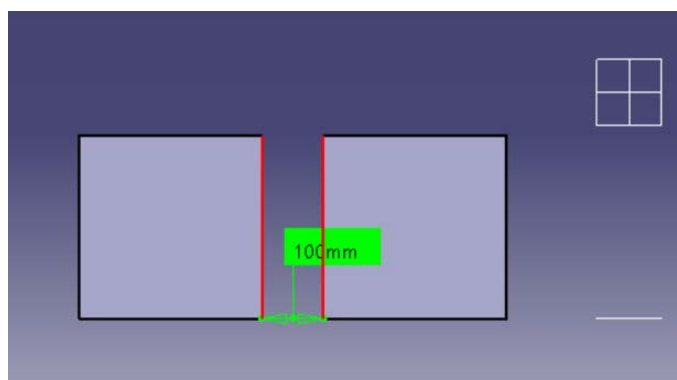


Figura 32. Separación entre asientos.

- Anchura del pasillo: identifica el valor introducido en el *TextBox* con la distancia considerada de izquierda a derecha o derecha a izquierda que hay entre los asientos interiores de la misma fila. La Figura 33 lo muestra.

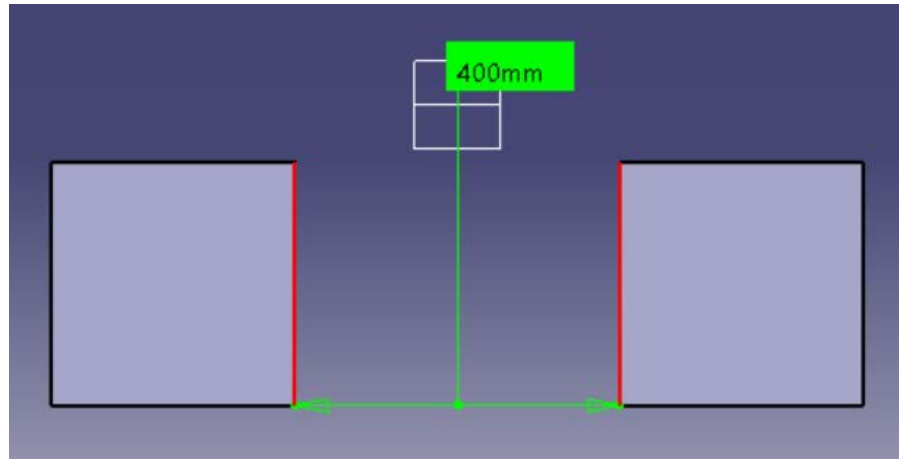


Figura 33. Anchura del pasillo.

Para la generación del conjunto de asientos se ha optado por seguir tres fases muy diferenciadas:

1. Tal y como se vió en la sección central, se va empezar con un *sketch* que va a permitir situar el asiento en la posición que el usuario ya ha definido en el formulario, después mediante el objeto *ShapeFactory*, se le hace un extrusionado con la herramienta *pad*.
2. A continuación, através del mismo objeto, se establecerá un *rectangular pattern* que desarrollará tanto el número de filas como el número de asientos que habrá en ellas.
3. A lo ya conseguido falta por añadir un *mirror* con respecto al plano XZ que origine los asientos que faltan y el pasillo como hueco que queda entre ambos grupos de asientos.
4. Por último se abrirá una ventana en la que se podrá proceder al guardado del *part*.

En definitiva, el código asociado al *CommandButton* correspondiente al pasaje de un pasillo generará automáticamente una pieza similar a ésta.

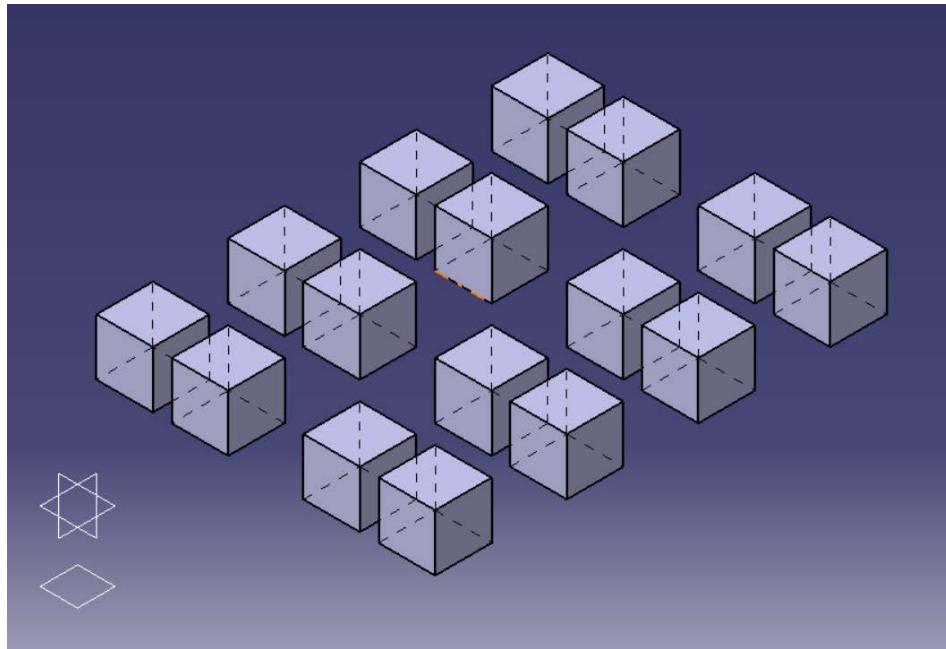


Figura 34. Pasaje de un pasillo.

7.3.2 Pasaje de dos pasillos

La forma de distribuir los asientos con dos pasillos no va a presentar mayores dificultades con respecto a la de un pasillo. Sí se puede decir que a la hora de poder desarrollarlo habrá diferencias entre una y otra. Cuando se accede al formulario a través del *CommandButton* se presenta lo siguiente:

Pasaje de dos pasillos ×

Offset delantero <input style="width: 60px;" type="text"/> mm	Anchura del asiento <input style="width: 60px;" type="text"/> mm
Número de filas <input style="width: 60px;" type="text"/>	Longitud del asiento <input style="width: 60px;" type="text"/> mm
Número de asientos por fila centrales <input style="width: 60px;" type="text"/>	Pitch <input style="width: 60px;" type="text"/> mm
Número de asientos por fila extremos <input style="width: 60px;" type="text"/>	Separación entre asientos <input style="width: 60px;" type="text"/> mm
Altura del asiento <input style="width: 60px;" type="text"/> mm	Anchura de los pasillos <input style="width: 60px;" type="text"/> mm

Figura 35. Formulario correspondiente al pasaje de dos pasillos.

Se observa que los datos necesarios son exactamente los mismos que los que se necesitaban en el anterior punto y sólo se realiza una distinción entre los asientos centrales y los que se encontrarán más cercanos a la estructura que denominaremos extremos. No conviene volver a repetir la definición de lo que representa cada *TextBox* ya que quedó identificado en el punto anterior, por tanto, se procederá a la forma de conseguir el conjunto de asientos. Los pasos que se van dando son:

1. Se comienza con un *sketch* que va a permitir situar el asiento central que tomaremos como referencia en la posición que el usuario ya ha definido en el formulario, para luego mediante el objeto *ShapeFactory*, hacerle un extrusionado con la herramienta *pad*.
2. A continuación, através del mismo objeto, se establecerá un *rectangular pattern* que desarrollará tanto el número de filas como el número de asientos centrales que habrá en ellas.
3. Una vez realizado esto, se inserta un *body* que va a servir para poder crear los asientos extremos siguiendo el mismo procedimiento para los asientos centrales.
4. Antes de pasar al guardado del *part*, se establece un *mirror* con respecto al plano XZ para poder hacer los asientos extremos del otro lado.
5. Se guarda el *part* correspondiente.

El resultado final se muestra a continuación en la Figura 34.

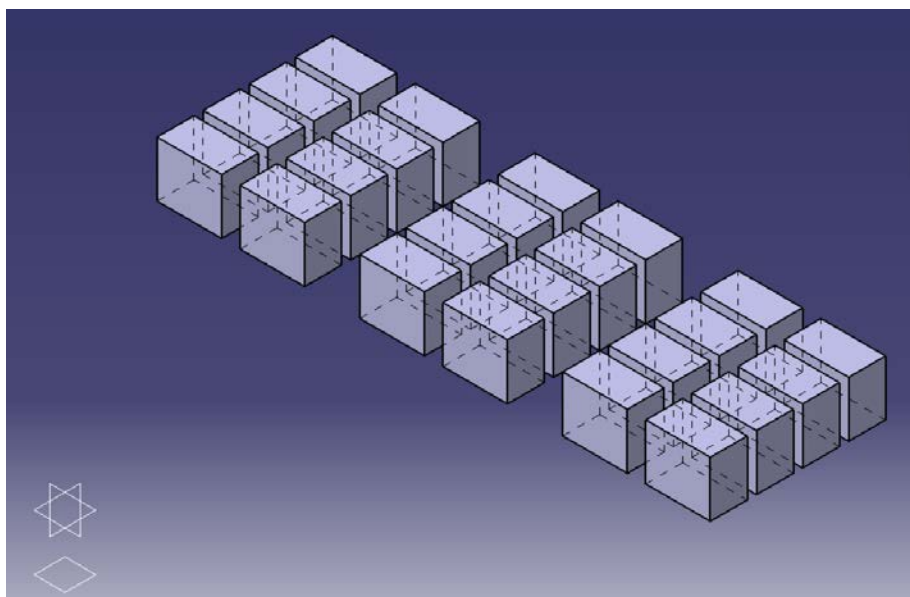


Figura 36. Pasaje de dos pasillos.

7.4 Pasajero

Siguiendo con la forma de proceder llevada a cabo hasta ahora, se accede a este formulario haciendo clic en el *CommandButton* correspondiente. La ventana que se abre es la siguiente:



The image shows a software window titled "Características del pasajero" with a close button (X) in the top right corner. The window contains three input fields for dimensions, each followed by "mm": "Altura", "Anchura", and "Longitud". Below these fields are two buttons: "Exportar datos" and "Atrás".

Figura 37. Formulario correspondiente a la definición del pasajero.

Tal y como se hizo con los asientos, se asimilará la forma del pasajero a la de un prisma para mayor facilidad de trabajo, por lo tanto, sólo tendremos que dar las tres dimensiones del pasajero medio que viaje. Como la definición de las tres dimensiones para este caso en particular resulta poco informativa, se continuará explicando el procedimiento llevado a cabo para que resulte el pasajero. Las etapas por las que se pasa para la generación del mismo son:

1. Se define un *sketch* con las dimensiones de anchura y longitud en planta que va a caracterizar al pasajero y después mediante el objeto *ShapeFactory*, se le hace un extrusionado con la herramienta *pad* a la altura elegida.
2. Después de esto se abrirá una ventana donde se podrá guardar este *part*.

Es importante indicar que el pasajero quedará situado justo en el centro de la cabina y a una distancia cero del plano YZ para tenerlo en cuenta por el usuario cuando se pase al análisis de las posibles interferencia. El resultado es el siguiente:

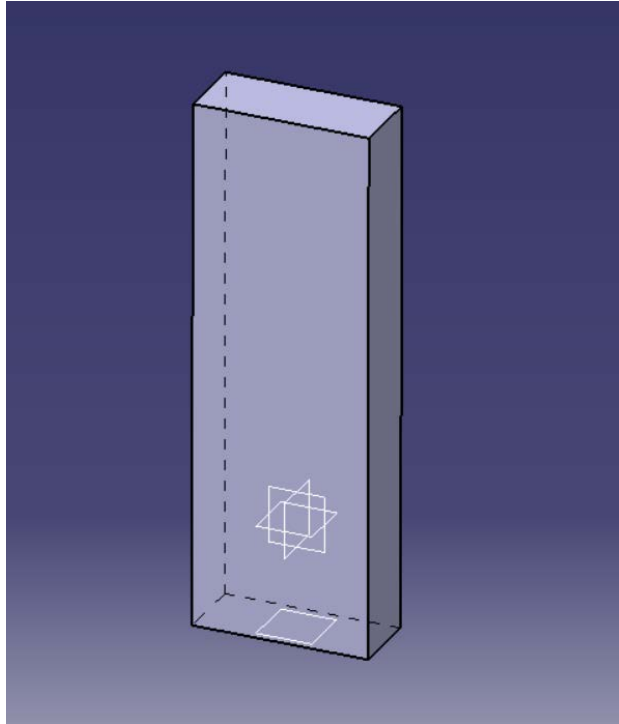


Figura 38. Pasajero.

7.5 Carga a transportar

El formulario que aparece tras hacer clic en el *CommandButton* de la carga a transportar de la ventana principal se muestra en la siguiente imagen:

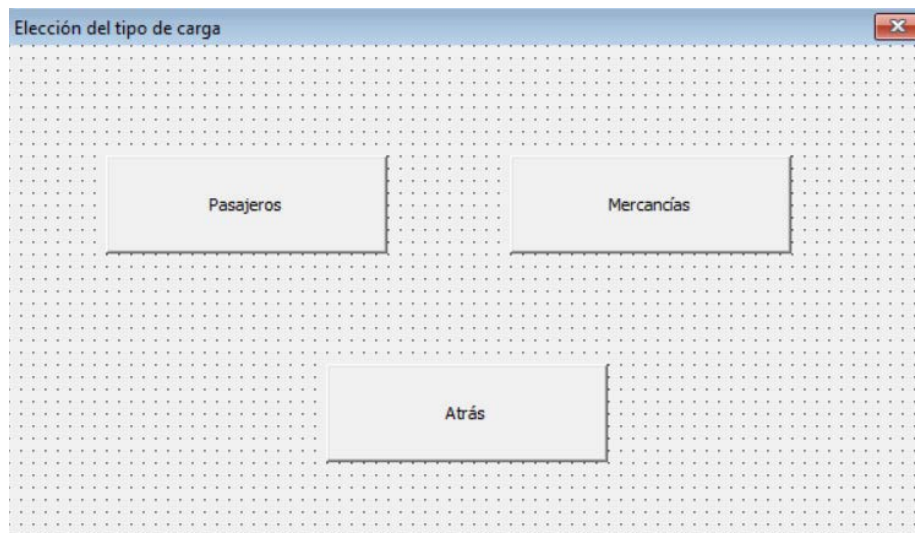


Figura 39. Formulario correspondiente a la elección del tipo de carga.

Tal y como se ve, se hace una distinción muy clara en lo que al transporte aéreo se refiere, es decir, diferenciar entre un avión de transporte de pasajeros u otro de

mercancías. Llegado a este punto, se debe ser lógico y consecuente si ya se había optado anteriormente por pasajeros, de forma que ahora se haga lo mismo. Cabe la posibilidad, por supuesto, de sustituir los estantes que disponen los pasajeros dentro del avión por transportar mercancía aunque a priori no parezca normal. Esta elección se deja a criterio del usuario.

7.5.1 Pasajeros

Si se escoge por pasajeros, se abre el siguiente formulario:

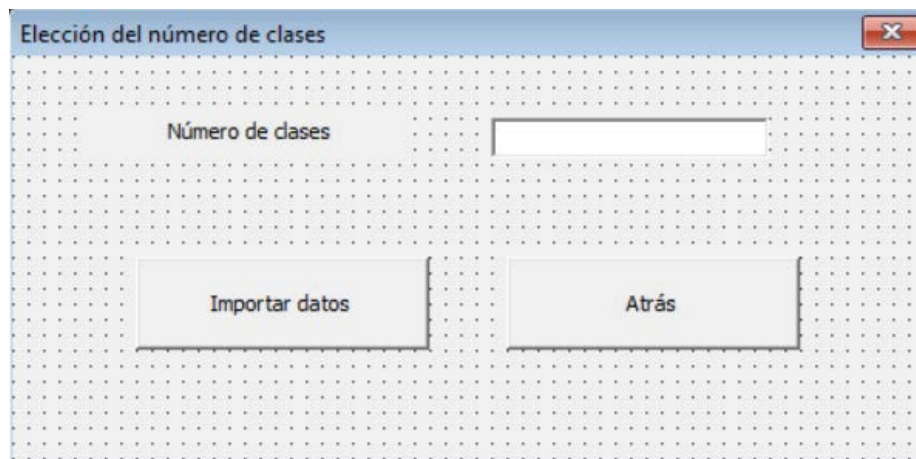
The image shows a software window titled "Elección del número de clases". It features a light blue header bar with a close button (X) on the right. The main area has a dotted background. At the top, the text "Número de clases" is followed by a white rectangular input field. Below this, there are two buttons: "Importar datos" on the left and "Atrás" on the right.

Figura 40. Formulario correspondiente a la elección del número de clases.

La apariencia y la función es la misma que el formulario que apareció cuando se definía el pasaje por lo que no conviene ahondar más en él. Cuando se hace clic para que se importe el número de clases elegidas, siendo congruente con lo ya definido en el pasaje, salta este otro formulario:

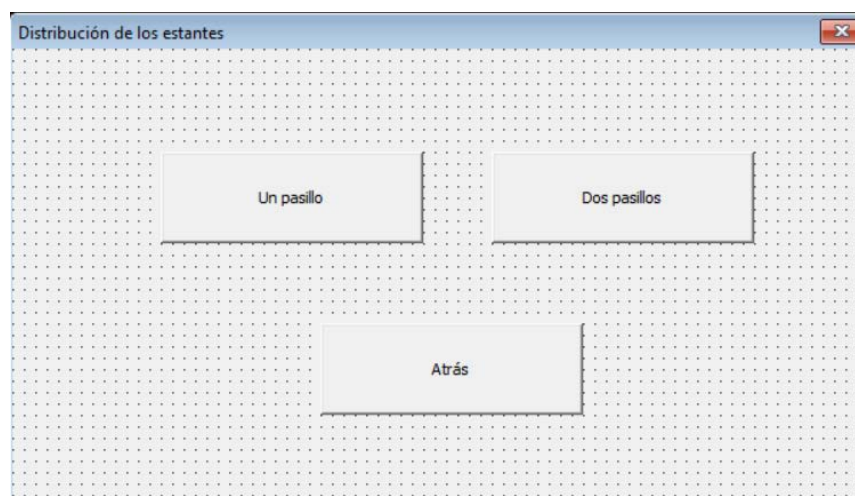
The image shows a software window titled "Distribución de los estantes". It has a light blue header bar with a close button (X) on the right. The main area has a dotted background. There are three buttons: "Un pasillo" on the left, "Dos pasillos" on the right, and "Atrás" centered below the other two.

Figura 41. Formulario correspondiente a la distribución de los estantes.

De nuevo, el formulario es el mismo y el objetivo que persigue también. Tal y como ocurrió con el caso del pasaje este formulario saldrá un número de veces igual al número de clases seleccionado en virtud del Código 28. Se recuerda al usuario que en este caso se está realizando la distribución de los estantes de equipaje que queda por encima de los asientos, de forma que sería coherente elegir el mismo número de clases para este caso. Se elija uno o dos pasillos las opciones de caracterización son las mismas para ambos casos de forma que la información quedará recogida en una ventana como ésta:



The image shows a software window titled "Estantes en distribución de un pasillo" with a close button (X) in the top right corner. The window has a light gray background and contains the following elements:

- A label "Offset delantero" followed by a text input field and the unit "mm".
- A label "Porcentaje de la altura del asiento por encima del mismo" followed by a text input field and the unit "%".
- A button labeled "Exportar datos" centered below the input fields.
- A button labeled "Atrás" centered below the "Exportar datos" button.

Figura 42. Formulario correspondiente a la carga de un pasillo.

Las variables de las que consta son:

- Offset delantero: identifica el valor introducido en el *TextBox* con la distancia desde el plano YZ al lugar en el cual se van a empezar a construir los estantes. Mismo concepto que el que ya se ha desarrollado anteriormente.
- Porcentaje de la altura del asiento por encima del mismo: identifica el valor introducido en el *TextBox* con la proporción de la altura de los asientos de más que poseerán los estantes. Proporcionará una medida del confort que tendrá el pasajero a la hora de salir del asiento. Tal y como se muestra en la Figura 43, el porcentaje sería del 20% para ese caso en particular. *Referencia [6]*.

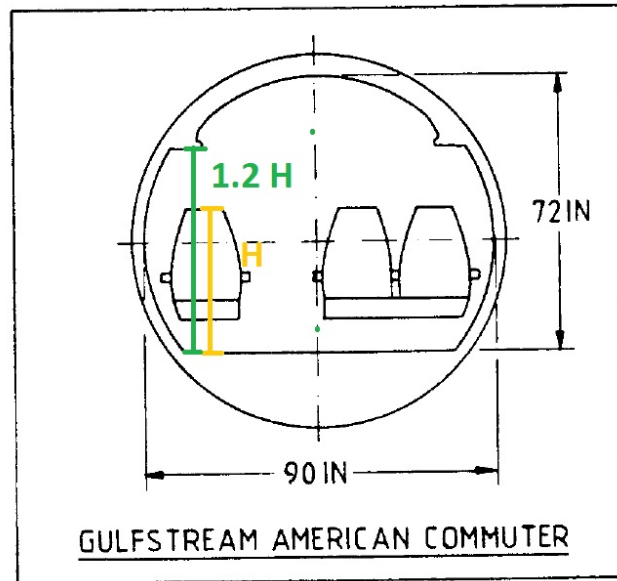


Figura 43. Porcentaje de la altura del asiento por encima del mismo.

El procedimiento para llevar a cabo la generación de los estantes tiene los siguientes pasos:

1. Para empezar se dibuja un *sketch* de una elipse que coincide con la elipse interior de las dos que definían a la sección central para luego mediante el objeto *ShapeFactory*, proceder a la extrusión con la herramienta *pad* en sentido inverso al que propone CATIA predeterminadamente, con la profundidad exigida.
2. A continuación, continuando con el objeto *ShapeFactory*, mediante la herramienta *pocket* se va a eliminar todo el material de la elipse que quede por debajo de la altura de referencia que no es otra que la del suelo de la cabina. Esto es:

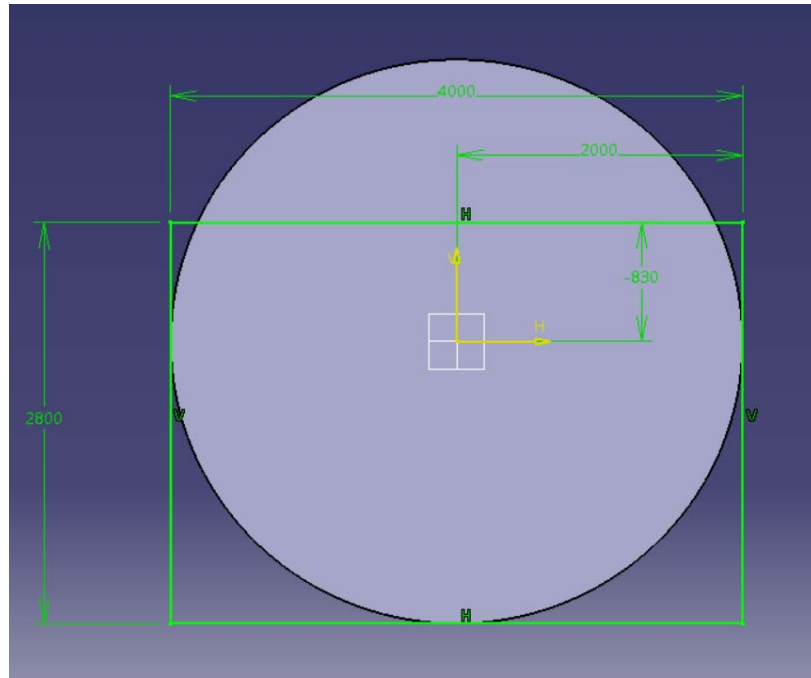


Figura 44. Restricción en altura.

Conviene decir que esta restricción de altura está condicionada por el valor de la altura del asiento.

3. De la misma forma, se eliminará todo el material de la elipse para poder crear el hueco que corresponde al pasillo. Este proceso se muestra a continuación:

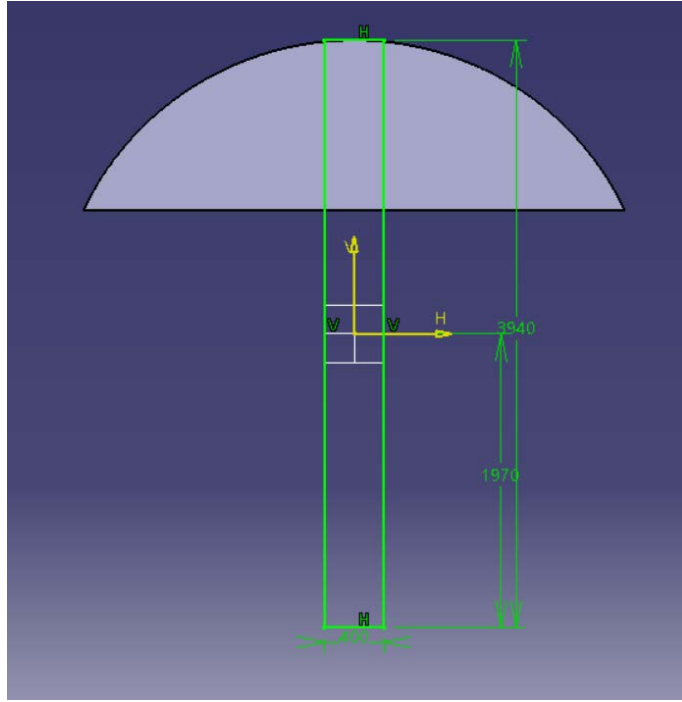


Figura 45. Hueco del pasillo.

4. Por último, se guarda el sólido resultante.

Para el caso de dos pasillos, el proceso es el mismo pero eliminando el material de la elipse correspondiente para generar el hueco de ambos pasillos. El resultado de todas estas operaciones se muestra tanto en la Figura 46 para el caso de un pasillo como en la Figura 47 para dos pasillos.

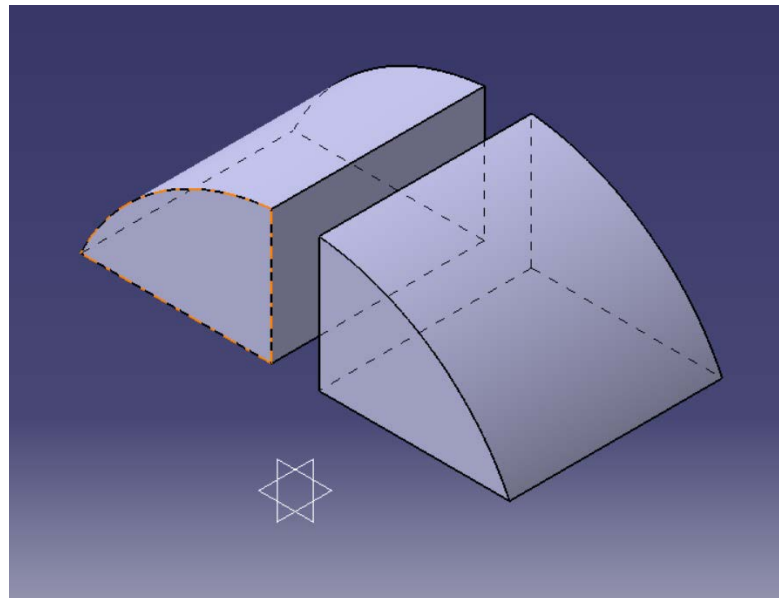


Figura 46. Estantes para un pasillo.

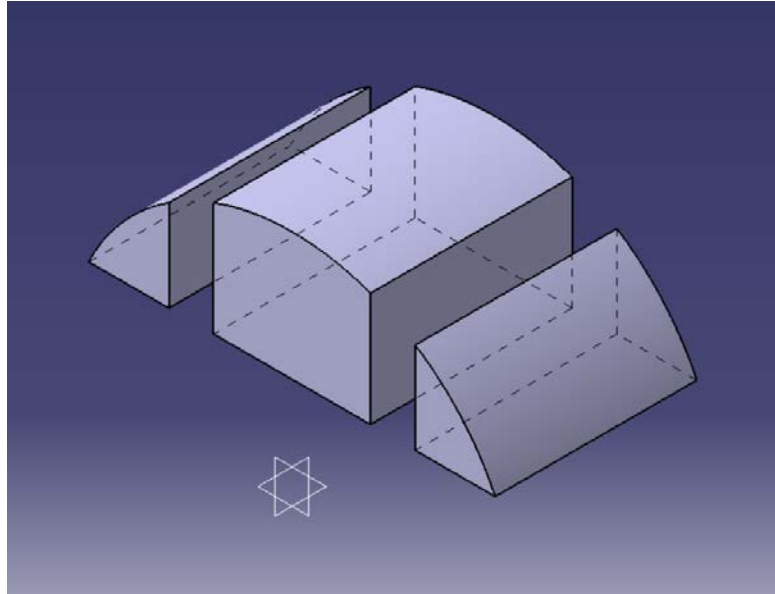


Figura 47. Estantes para dos pasillos.

7.5.2 Mercancía

Si se escoge por pasajeros, se abre el siguiente formulario:

Dimensiones de la mercancía ×

Offset delantero	<input type="text"/>	mm
Altura	<input type="text"/>	mm
Anchura	<input type="text"/>	mm
Longitud	<input type="text"/>	mm

Figura 48. Formulario correspondiente a las dimensiones de la mercancía.

Si se pretende llevar mercancías en el avión se puede optar a ello, eso sí de la forma más sencilla posible, desde este punto. En la ventana se va a definir un prisma de unas determinadas dimensiones elegidas por el usuario. Además, se podrá escoger dónde

sitarlo a través del offset delantero. Dicha mercancía quedará situada justo en el centro de la cabina teniendo el plano XZ de simetría. La forma de proceder sería la siguiente:

1. Se define un *sketch* con las dimensiones de anchura y altura en el plano YZ y después mediante el objeto *ShapeFactory*, se le hace un extrusionado con la herramienta *pad* con la profundidad deseada.
2. Después de esto se abrirá una ventana donde se podrá guardar este *part*.

El resultado sería lo que se aprecia en la Figura 49:

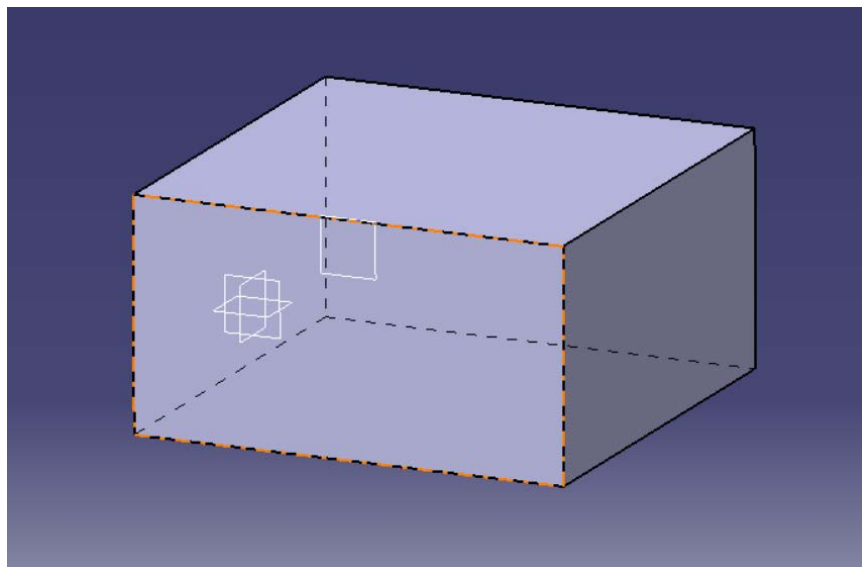


Figura 49. Mercancía.

7.6 Carga en la bodega

El acceso a este formulario se hace como los ya comentados hasta ahora, a través del principal pulsando su *CommandButton*. Inmediatamente salta la siguiente ventana:

Dimensiones de la carga en la bodega X

Altura menor mm

Altura mayor mm

Base menor mm

Base mayor mm

Profundidad mm

Exportar datos

Atrás

Figura 50. Formulario correspondiente a la definición de la carga.

En el aparecen aparecen cinco datos geométricos para crear la carga que se desea. Resulta imprescindible comentar que de entre la inmensa variedad de containers y pallets que existen, una muestra de ello se observa en las Figuras 51 y 52, se decidió elegir el modelo correspondiente al Boeing 727-200C dentro de los *Belly Containers* al ser representativo de lo que se pretende conseguir y tener cierta cierta dificultad de programación, al menos en comparación con el resto (Figura 53). Referencia [7] y [8].

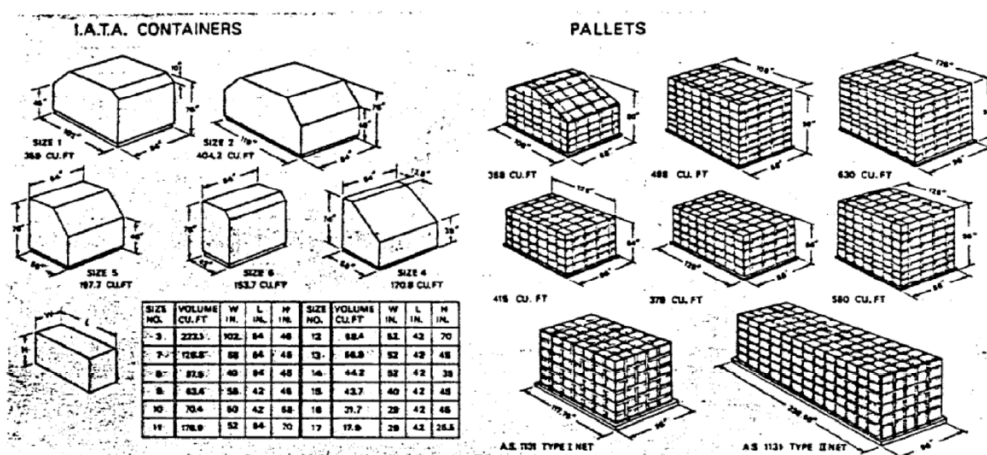


Figura 51. Disposición de carga I.

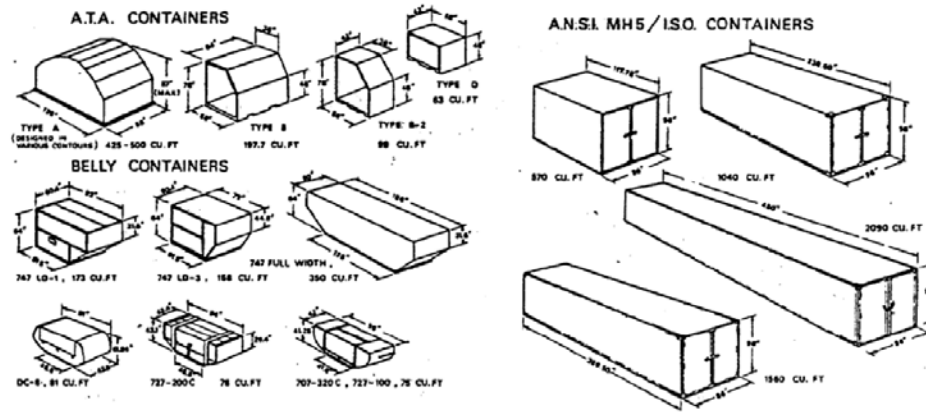


Figura 52. Disposición de carga II.

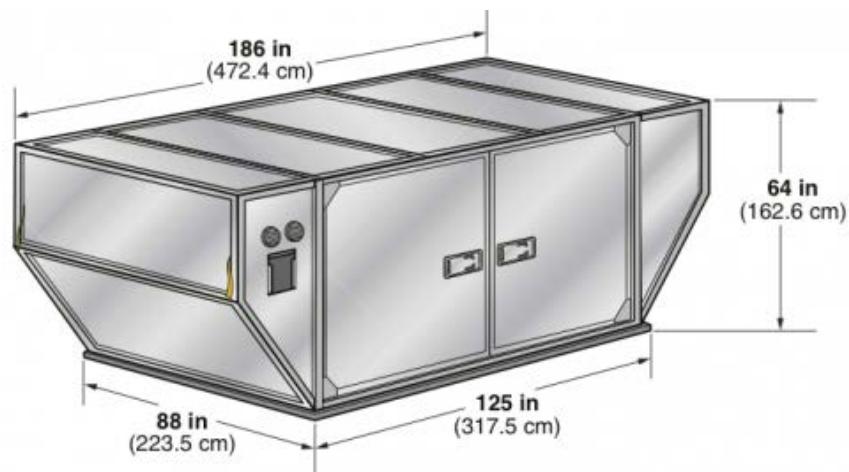


Figura 53. Modelo elegido.

La forma del mismo se puede asemejar a un trapecio, que no es más que un cuadrilátero que tiene dos lados no consecutivos paralelos llamados bases del trapecio y la distancia entre ellos altura. Sin más, esos son los datos que hay que proporcionar aparte de la profundidad a la que hay que extender el objeto. Se dirá además que el container quedará situado justo en la zona inferior de las dos en las que la cabina queda dividida por el suelo pegado al mismo y a una distancia cero del plano YZ. Las fases del proceso se detallan ahora:

1. Se define un *sketch* con la mitad de las dimensiones de las bases y las alturas que van a caracterizar a medio container y después mediante el objeto *ShapeFactory*, se le hace una extrusión con la herramienta *pad* a la profundidad elegida.
2. Por último, se guarda el *part* correspondiente.

El resultado queda visible en la siguiente figura:

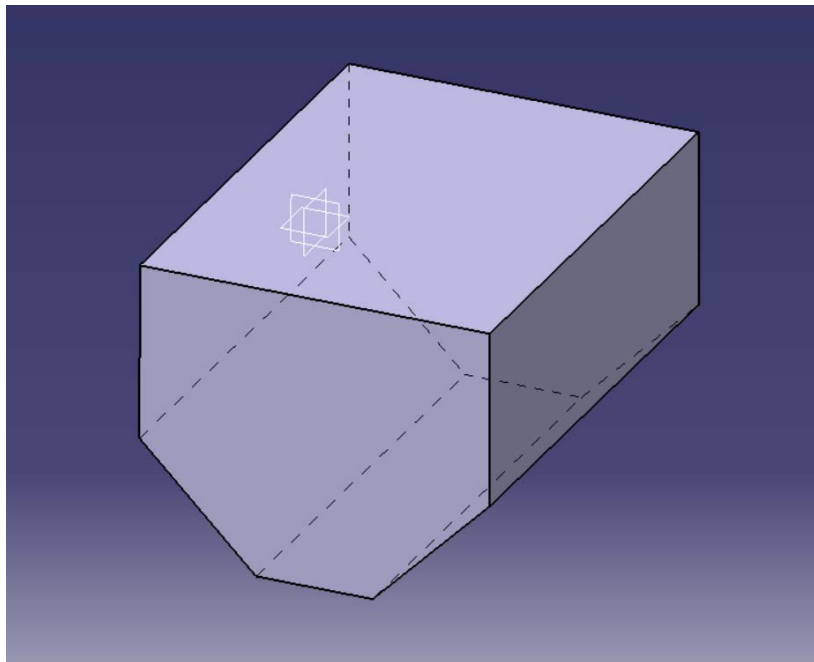


Figura 54. Carga en la bodega.

7.7 Baño

Una vez más desde el formulario principal se accede al del baño pulsando el *CommandButton* asociado. La ventana que aparece a continuación del mismo es la que se presenta en la Figura 55:

A screenshot of a software dialog box titled "Restricciones del baño" (Bathroom Restrictions). The dialog box has a close button (X) in the top right corner. It contains three input fields, each followed by the unit "mm":

- Offset delantero
- Anchura del baño
- Profundidad

Below the input fields are two buttons: "Exportar datos" and "Atrás".

Figura 55. Formulario correspondiente a la definición del baño.

En él se observa la necesidad de introducir tres datos para que se pueda dar la definición completa del mismo. Estos datos son:

- Offset delantero: identifica el valor introducido en el *TextBox* con la distancia desde el plano YZ al lugar en el cual se va a empezar a construir el baño. Nótese que la asignación del offset no es más que una sencilla forma de situar al baño dentro de la cabina.
- Anchura del baño: identifica el valor introducido en el *TextBox* con la dimensión del baño considerada de derecha a izquierda o de izquierda a derecha, en contraposición a la considerada de arriba abajo o de abajo arriba en una superficie. La anchura del baño se empezará a medir desde el radio mayor de la elipse interior hacia dentro. Una explicación a esta definición se considerará más adelante a la hora de explicar el procedimiento desarrollado.
- Profundidad: identifica el valor introducido en el *TextBox* con la extensión a la que se va a extruir el baño.

La forma que se adoptó para poder dar forma al baño, y la que se cree que es más sencilla, consiste en construir una superficie que ocupe completamente el interior de la zona superior de la cabina, habiéndole impuesto previamente una restricción en altura, extruirla a la profundidad deseada e imponerle una restricción en anchura. No se está diseñando el baño como tal, pero si se está rellenando el espacio que ocuparía el mismo con las dimensiones establecidas. El procedimiento comentado es el siguiente:

1. Para empezar se dibuja un *sketch* de una elipse que coincide con la elipse interior de las dos que definían a la sección central para luego mediante el objeto *ShapeFactory*, proceder a la extrusión con la herramienta *pad* en sentido inverso al que propone CATIA predeterminadamente, con la profundidad exigida.
2. A continuación, continuando con el objeto *ShapeFactory*, mediante la herramienta *pocket* se va a eliminar todo el material de la elipse que quede por debajo de la altura de referencia que no es otra que la del suelo de la cabina. Esto es:

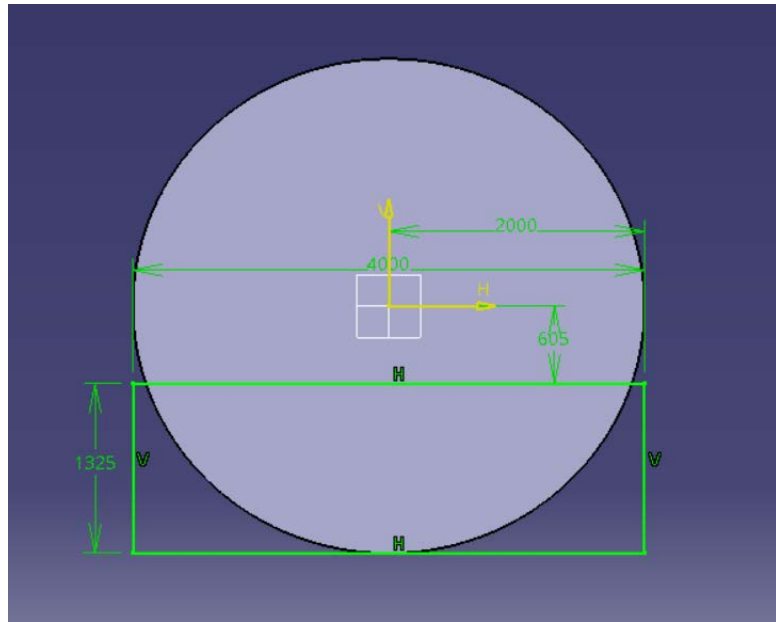


Figura 56. Restricción en altura.

Conviene decir que esta restricción de altura está supeditada al valor de la altura del suelo que se impuso cuando se creó la sección central del fuselaje.

3. De la misma forma, se eliminará todo el material de la elipse que quede a la derecha de la anchura del baño que se ha exigido. Este proceso se muestra a continuación:

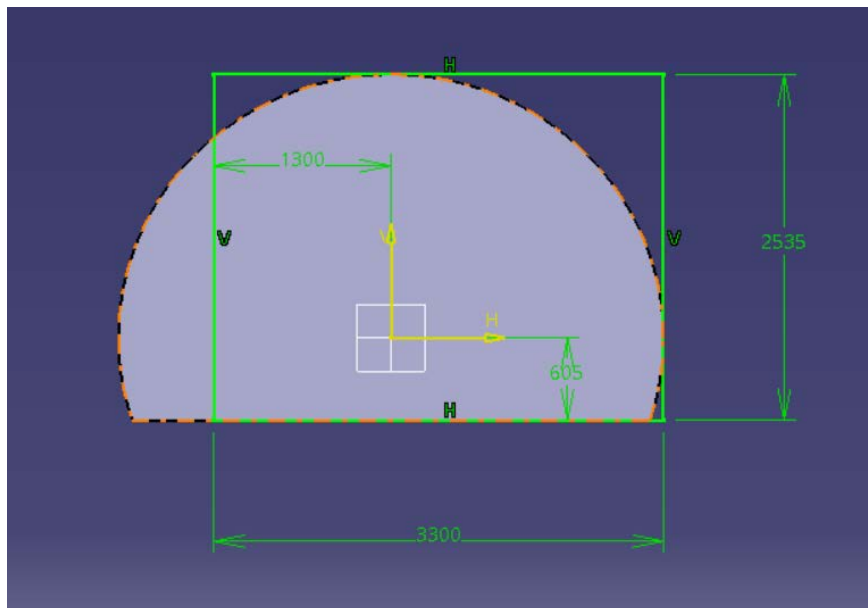


Figura 57. Restricción en anchura.

4. Por último, y tal y como se ha ido haciendo hasta ahora, se guarda el sólido resultante.

El resultado de todas estas operaciones se muestra en la Figura 58.

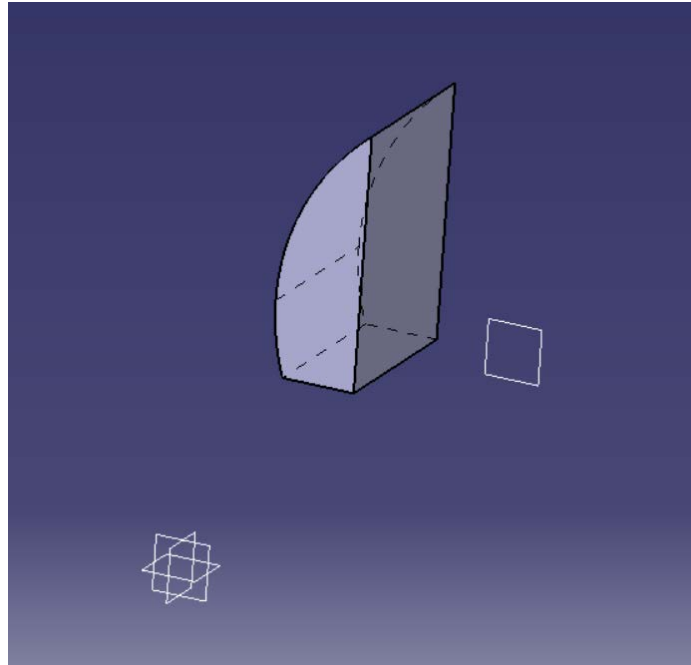


Figura 58. Baño.

7.8 Galley

El galley no es más que la cocina de los aviones, donde se guardan las comidas y bebidas porque realmente allí no se cocina nada. Su formulario presenta la forma siguiente:

Restricciones del galley

Offset delantero mm

Profundidad mm

Exportar datos

Atrás

Figura 59. Formulario correspondiente a la definición del galley.

Sólo dos datos se requieren para poder dar forma al galley, tanto el offset delantero como la profundidad poseen la misma definición que en el caso del baño por lo que se puede ir directamente al proceso. Siguiendo el mismo razonamiento que para el caso del baño basta con construir una superficie que ocupe completamente el interior de la zona superior de la cabina, habiéndole impuesto previamente una restricción en altura y extruirla a la profundidad deseada. Para el caso del galley no hay restricción alguna en anchura. El procedimiento es como sigue:

1. Para empezar se dibuja un *sketch* de una elipse que coincide con la elipse interior de las dos que definían a la sección central para luego mediante el objeto *ShapeFactory*, proceder a la extrusión con la herramienta *pad* en sentido inverso al que propone CATIA predeterminadamente, con la profundidad exigida.
2. A continuación, continuando con el objeto *ShapeFactory*, mediante la herramienta *pocket* se va a eliminar todo el material de la elipse que quede por debajo de la altura de referencia que no es otra que la del suelo de la cabina. Esto es:

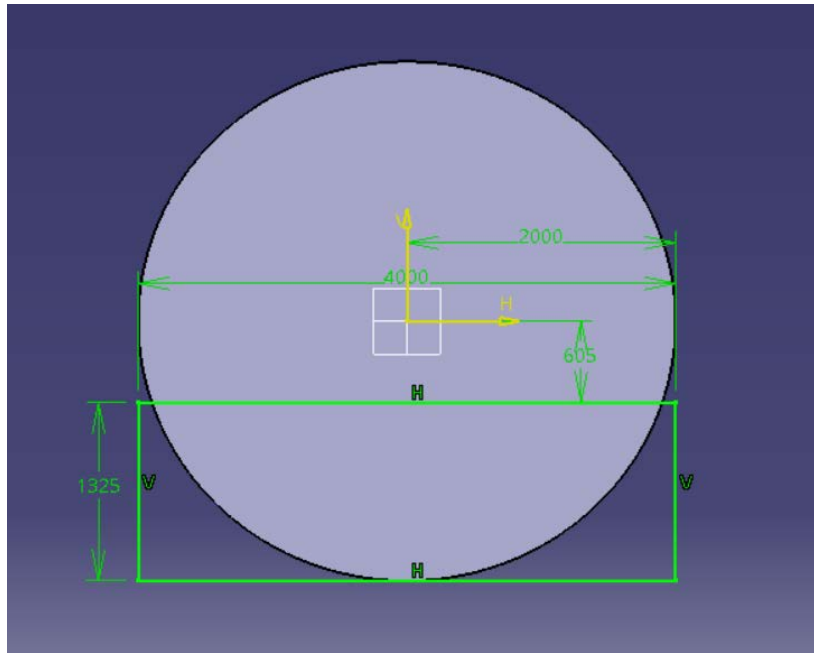


Figura 60. Restricción en altura.

De la misma manera que para el baño, la restricción de altura está supeditada al valor de la altura del suelo que se impuso cuando se creó la sección central del fuselaje. El resultado es éste:

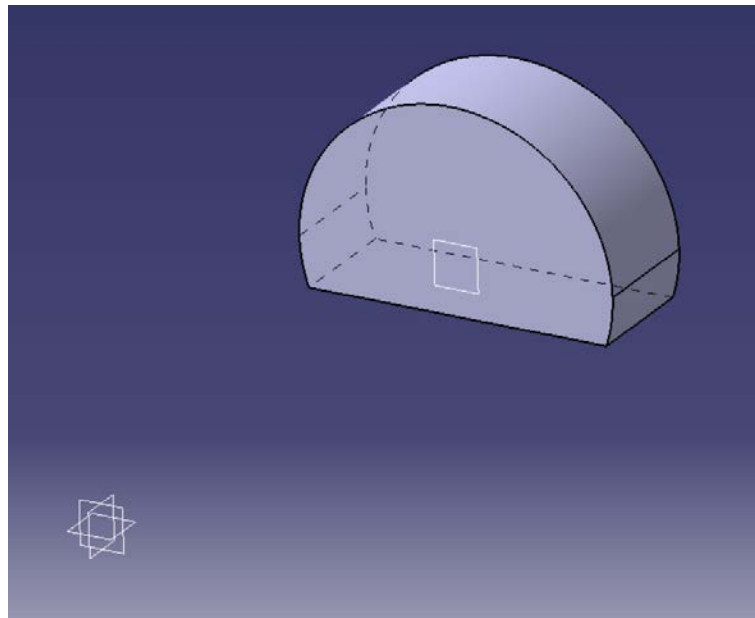
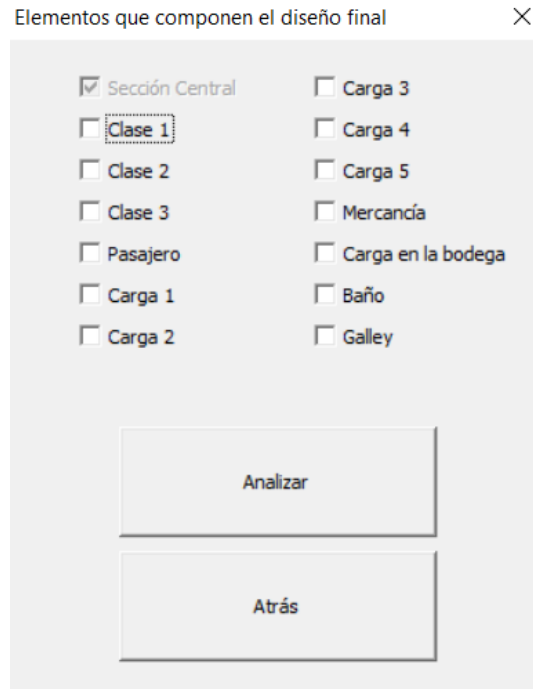


Figura 61. Galley.

7.9 Product

Definidas todas las piezas sólo quedaría hacer clic en el *CommandButton* desde la ventana principal para a continuación poder elegir las piezas que formaran parte de nuestro *Product* final. Este procedimiento se selecciona a través de la siguiente ventana:



Elementos que componen el diseño final

<input checked="" type="checkbox"/> Sección Central	<input type="checkbox"/> Carga 3
<input type="checkbox"/> Clase 1	<input type="checkbox"/> Carga 4
<input type="checkbox"/> Clase 2	<input type="checkbox"/> Carga 5
<input type="checkbox"/> Clase 3	<input type="checkbox"/> Mercancía
<input type="checkbox"/> Pasajero	<input type="checkbox"/> Carga en la bodega
<input type="checkbox"/> Carga 1	<input type="checkbox"/> Baño
<input type="checkbox"/> Carga 2	<input type="checkbox"/> Galley

Analizar

Atrás

Figura 62. Formulario correspondiente a la elección de *parts*.

En él se puede ver la posibilidad de seleccionar entre tres clases, cinco cargas, pasajero, mercancías, carga en la bodega, baño y galley. La opción de la sección central se encuentra marcada y bloqueada pues de su existencia depende que el proceso termine satisfactoriamente, por lo que el usuario previamente tendría que haberla definido. Una vez que se eligen las piezas que se desea analizar, se presiona el *CommandButton* y se espera hasta que el proceso finalice.

A los resultados de ese análisis se puede acceder de dos formas, o bien directamente desde el entorno de trabajo, más concretamente desplegando continuamente la rama del árbol de navegación denominada *Applications* hasta llegar a los resultados o bien a través de un archivo *.xml* que se exporta automáticamente mediante la programación. Se explicará con más detalle ambos casos a continuación.

7.9.1 Análisis de resultados desde el árbol de navegación

Tal y como se muestra en la siguiente imagen, llegar a los resultados por este método es sencillo.

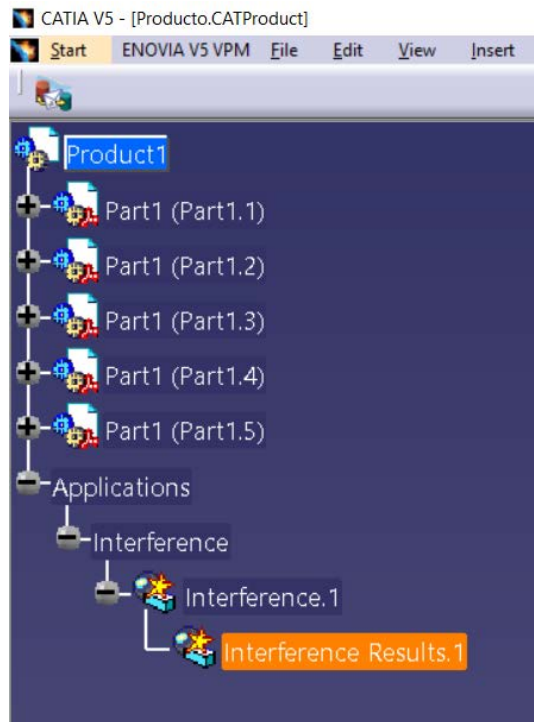


Figura 63. Árbol de navegación.

Haciendo doble clic en *Interference Results.1* se nos abren dos ventanas en las que por un lado estarán clasificadas las distintas interferencias y por otro lado podremos visualizarlas. En la siguiente figura se puede ver la lista de interferencias encontradas, que para este caso objeto de estudio han sido dos choques (*clash*), un contacto (*contact*) y ninguna *clearance*.

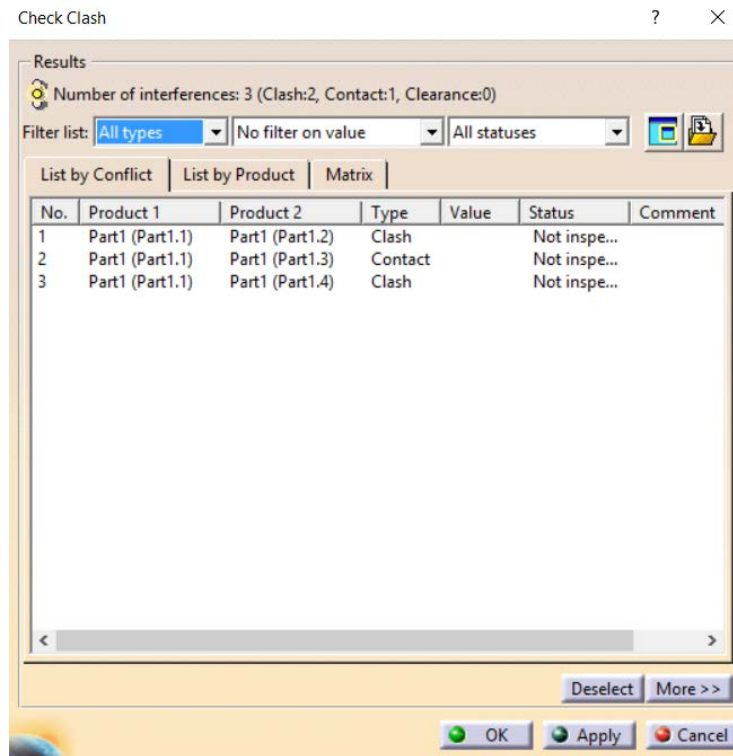


Figura 64. Ventana correspondiente a los resultados del análisis.

La misma te permite multitud de opciones tales como filtrar la lista por tipos de interferencia, ordenarla de valores mayores a menores de contacto y viceversa y filtrarlas en función del estado, ya sea no inspeccionado, relevante o irrelevante. Además a partir de ella puedes exportar estos resultados a otros formatos como *.xml*, *.txt*, *.model* o *.cgr*. Otra de las funciones que permite esta ventana es la posibilidad de visualizar las interferencias que existen sin más que seleccionarlas en la tabla, apareciendo en color rojo los choques y en amarillo los contactos. Una muestra de ello se puede ver en la Figura 65.

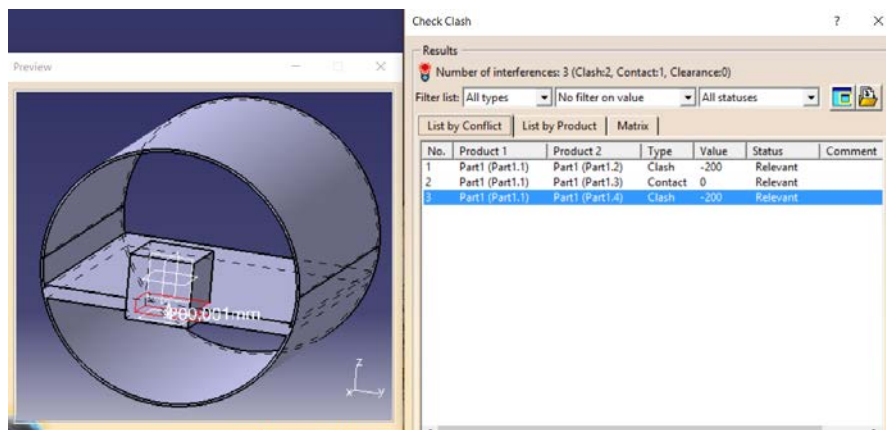


Figura 65. Ejemplo.

Gracias a esta tabla el usuario podrá corregir los posibles fallos que haya tenido a la hora del diseño. Cabe decir que la opción de exportar la tabla a diferentes formatos es muy interesante, sobre todo para poder llevar un control de lo que se va realizando. En el siguiente apartado hablaremos del formato *.xml* en concreto.

7.9.2 Análisis de resultados a través de un archivo *.xml*

Se ha elegido este formato puesto que CATIA permite hacerlo a través del objeto *CatClashExportTypeXMLResultOnly*, con el siguiente bloque de programación:

```
Dim Clashes As Clashes
Set Clashes = CATIA.ActiveDocument.Product.GetTechnologicalObject("Clashes")

Dim Clash As Clash
Set Clash = Clashes.Add()

Clash.ComputationType = catClashComputationTypeBetweenAll
Clash.Compute
Clash.Export CatClashExportTypeXMLResultOnly, Ruta & "\Análisis.xml"
```

Código 29. Bloque que permite exportar el análisis en archivo *.xml*.

El archivo *.xml* se guardará en la carpeta donde se hayan guardado todas las piezas. El mismo vendrá con otras carpetas y archivos que son necesarios para poder abrirlo. Existen multitud de formas con las que abrir este documento:

- Usando un editor de texto: el archivo se puede abrir usando por ejemplo el Bloc de notas. No obstante, no es una opción recomendada puesto que resulta bastante ilegible y habría que saber interpretar el texto.
- Utilizando un navegador web: si se abre con un navegador web será más fácil explorarlo. Esto se debe a que la mayoría de los navegadores aplican sangrías en las etiquetas automáticamente y te permiten contraer o expandir cada sección del árbol XML. Se ha comprobado que funciona correctamente tanto en Internet Explorer como en Mozilla Firefox. Esta es la opción más interesante puesto que resulta muy visual. En ella se pueden distinguir tanto las interferencias como los resultados de la computación. En *Interference.1* se muestran los *parts* componentes y la forma de análisis, en este caso, existen tanto contactos como choques. En *Computation Result* aparecen los componentes afectados y unos enlaces

que nos permiten ver cada caso por separado. Mostramos un ejemplo en la Figura 66, 67 y 68.

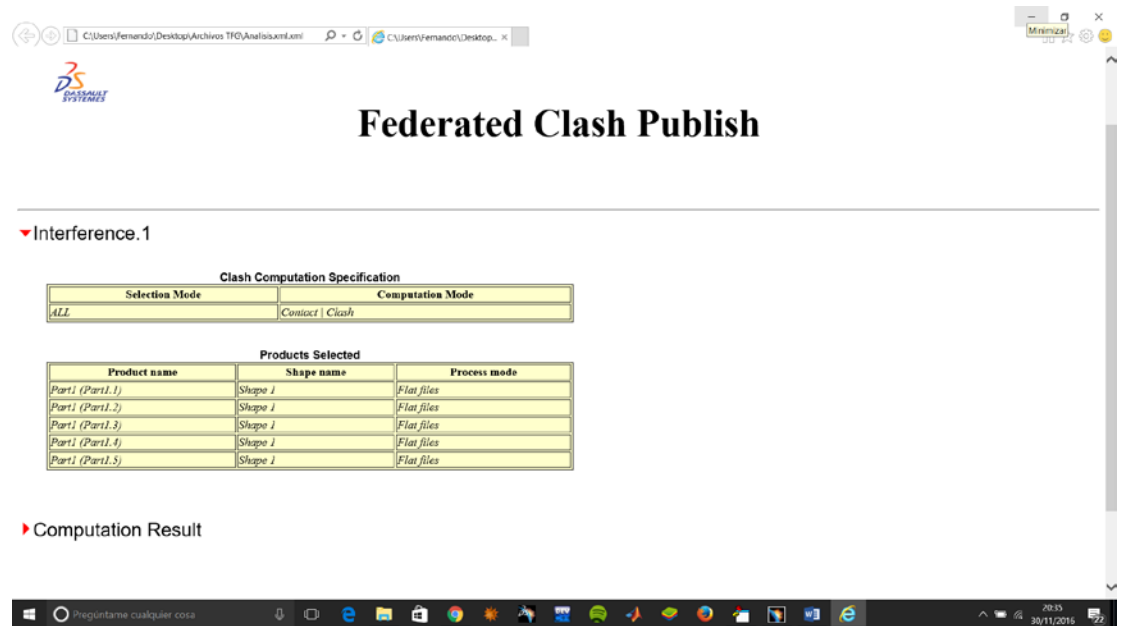


Figura 66. Abierto con navegador web I.



Figura 67. Abierto con navegador web II.

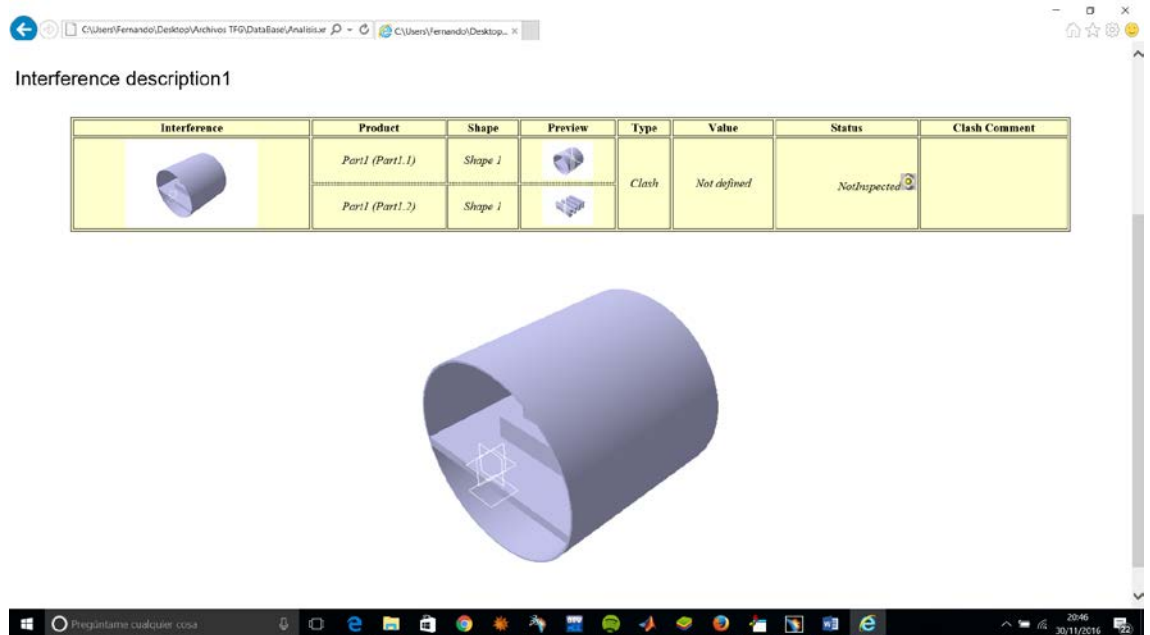


Figura 68. Descripción de la interferencia tomada como ejemplo.

- Usando Excel: Excel puede convertir un archivo XML en una tabla, lo cual puede ser bastante útil para procesar información pero queda restringido a que el usuario lo tenga instalado.
- Empleando algún visor XML

Sin duda queda patente que la forma más cómoda, y por otro lado la que el autor recomienda, es abrirlo con el navegador. Por último quedaría mostrar un *Product* como ejemplo, que se obtuviese tras ensamblar las piezas elegidas. Para mayor claridad en la visualización se ha ocultado la sección central. En la Figura 69, 70 y 71 se aprecian dos clases, una de un pasillo y otra de dos, así como sus correspondientes estantes, también el pasajero y la carga en la bodega, ambos en la parte frontal y por último en la parte posterior tanto el baño como el galley.

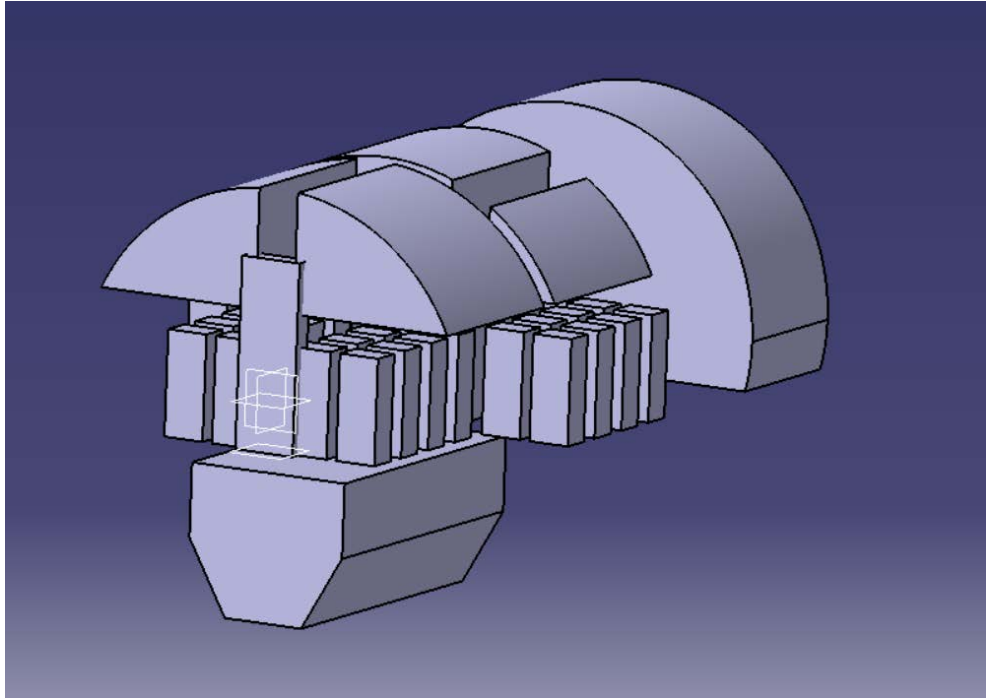


Figura 69. Ejemplo completo I.

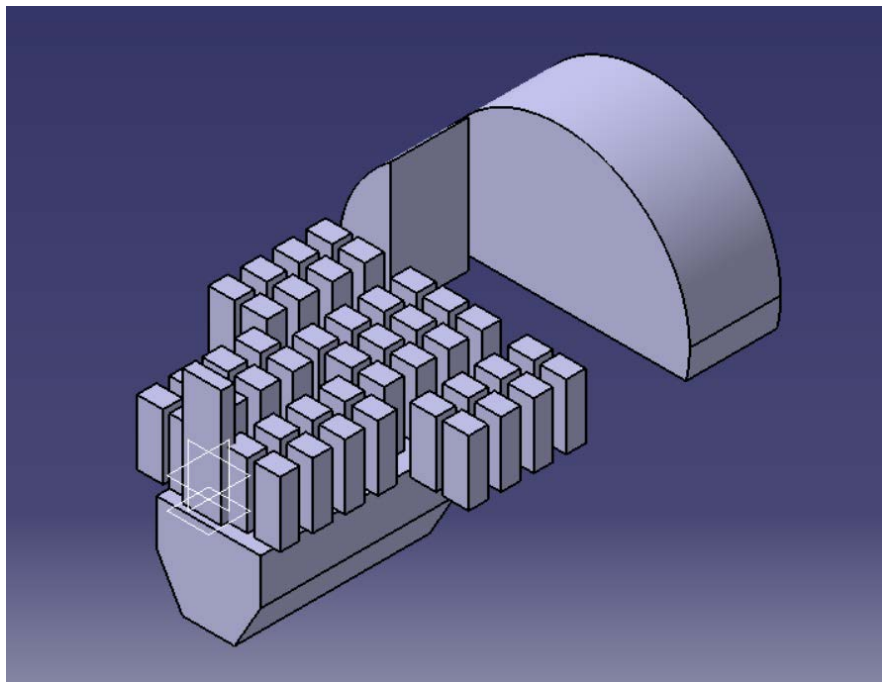


Figura 70. Ejemplo completo II.

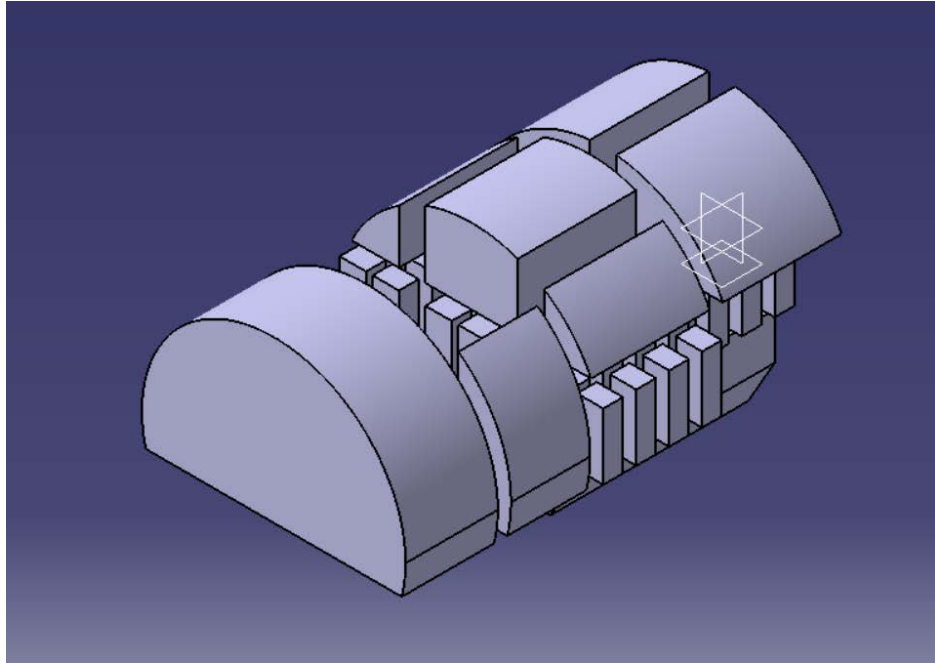


Figura 71. Ejemplo completo III.

8. Aplicación Diseño de Interiores: manual de usuario

Una vez explicado el software desarrollado para realizar las diferentes piezas, se va a ilustrar a continuación los pasos que debe seguir el lector para el uso del mismo.

En primer lugar, el usuario debe arrancar CATIA V5. Debe abrir la pestaña *Tools* de la barra de herramientas y acceder a las macros → *Macros....* Al hacer clic le aparecerá la ventana siguiente:

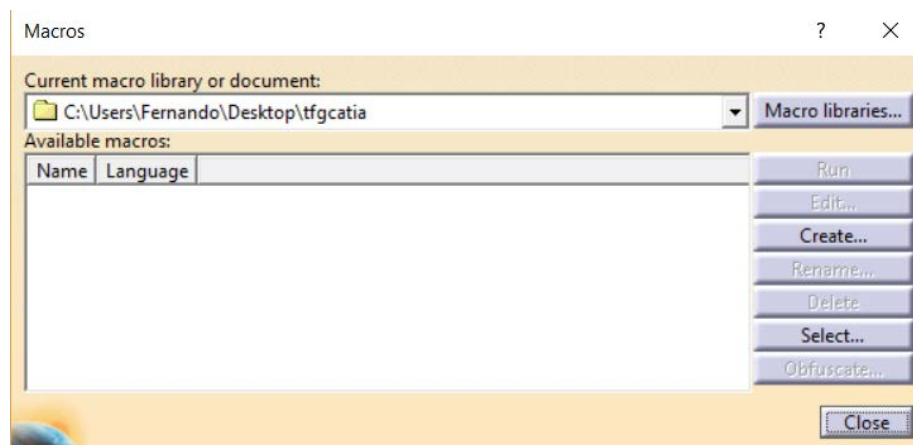


Figura 72. Biblioteca actual de macros.

Desde ella se accederá a la librería de macros donde se encuentran tanto el proyecto como los formularios necesarios para que la aplicación funcione correctamente, clicando primeramente en *Macro libraries...*, evento que hará aparecer otra ventana en la que clicando sobre el desplegable se seleccionará *Directories* pues será una carpeta existente en la que se tendrán los archivos. El procedimiento se muestra en las Figuras 73 y 74.

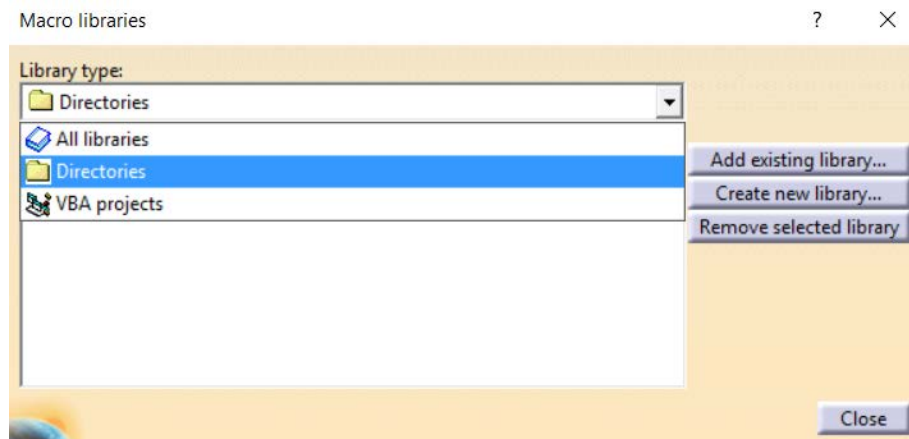


Figura 73. Pantalla de selección de la librería de macros.

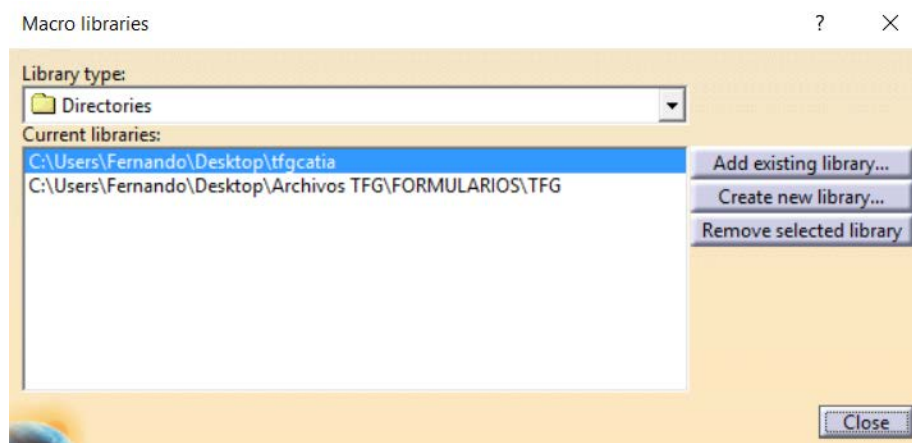


Figura 74. Pantalla de selección del directorio.

En ella misma se clicará en *Add existing library...* para seleccionar la carpeta en donde se hayan guardado los archivos anteriormente comentados, la cual estará en el lugar que el usuario haya decidido previamente. Una vez localizada la carpeta, cerramos la ventana *Macro libraries*.

En la ventana *Macros* se hace clic en *Select...* para seleccionar el proyecto *.catvba* que hará posible la ejecución de la aplicación, teniendo en cuenta que hay que cambiar el tipo de archivo a visualizar tal y como se muestra en las Figuras 75 y 76. Se abre el proyecto y se cierra la ventana *Macros*.

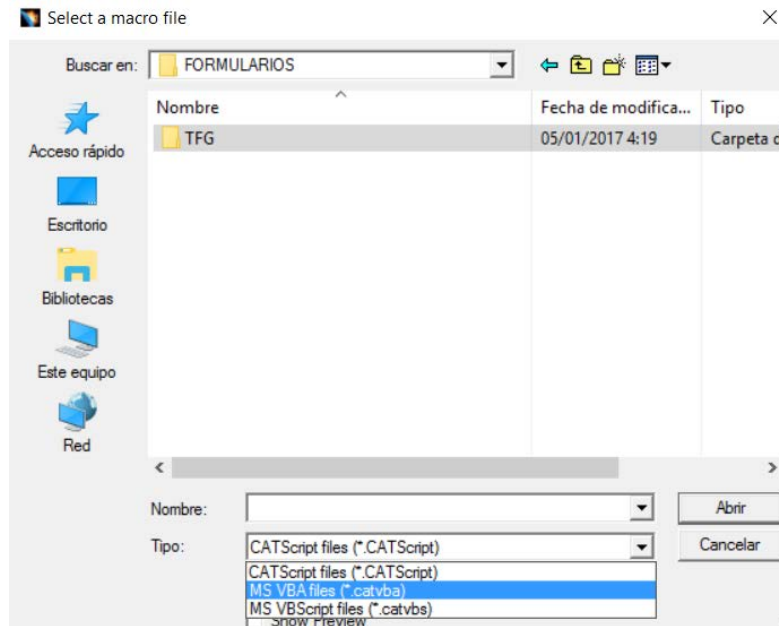


Figura 75. Cambio del tipo de archivo a visualizar.

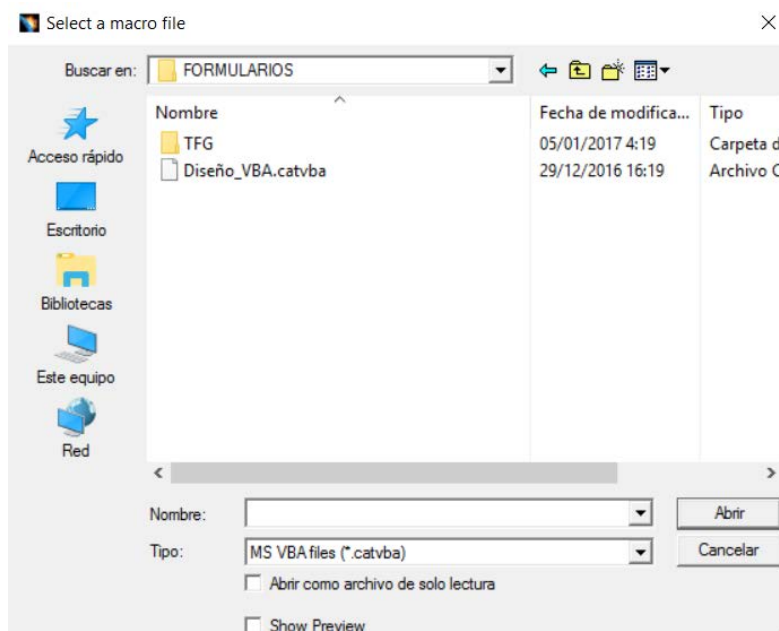


Figura 76. Selección del proyecto .catvba.

Una vez abierto la interfaz de CATIA, a través de la pestaña *Tools* de la barra de herramientas se accede al editor de Visual Basic haciendo clic en *Macros* → *Visual Basic Editor*.... Para arrancar la macro se debe clicar sobre el formulario “VentanaPrincipal” y reproducirla dándole al botón *play* de la barra de herramientas. Se abrirá la siguiente ventana:

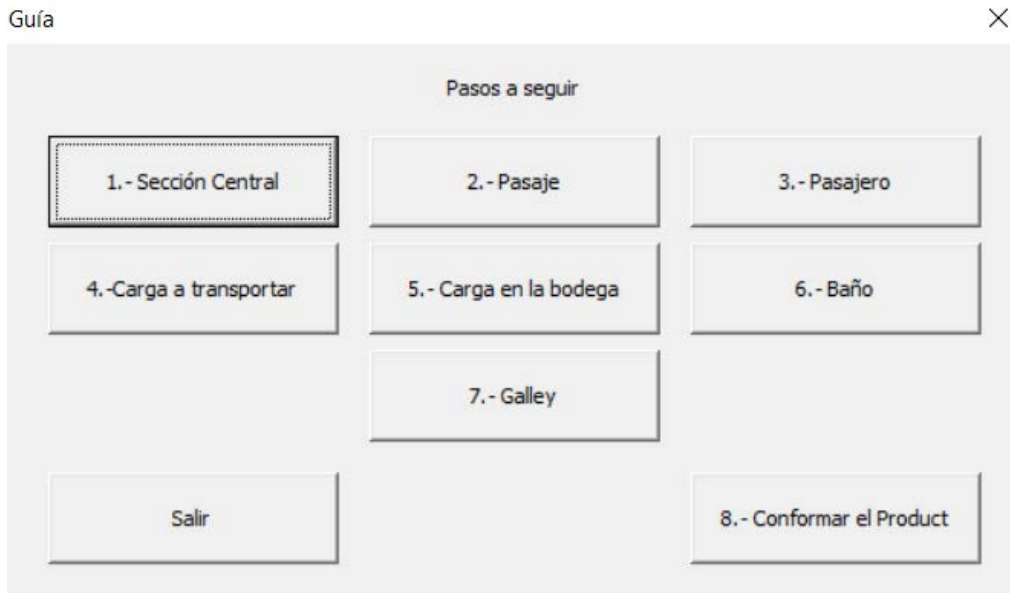


Figura 77. Interfaz de la aplicación Diseño de Interiores.

Sobre esta ventana se irá seleccionando las piezas que se quieren ir generando teniendo en cuenta ciertas limitaciones ya comentadas anteriormente pero que conviene recordar para el buen funcionamiento de la interfaz. Por un lado, es imprescindible generar la sección central, pues determinados datos como el radio mayor y menor de la elipse interior que la conforman y la altura del suelo son empleados en otros formularios para la correcta colocación de la pieza que se esté definiendo en ese momento. Como existe herencia de datos entre unos formularios y otros, y no es permanente, conviene no saltar entre formularios sin seguir el orden descrito, de lo contrario la aplicación no funcionará correctamente pues habrá datos que no estén definidos provocando el error y que tengamos que definir las piezas de nuevo. Se ilustrará a continuación un ejemplo sencillo.

Tal y como se ha dicho el primer paso sería definir las características de la sección central, por lo que para ello basta con hacer clic en su correspondiente botón.

The image shows a software window titled "Características de la sección central" with a close button (X) in the top right corner. The window contains a form with the following fields and values:

Característica	Valor	Unidad
Radio Mayor	2000	mm
Radio Menor	1970	mm
Espesor	30	mm
Altura del suelo	1300	mm
Espesor del suelo	100	mm
Profundidad	9000	mm

Below the input fields, there are two buttons: "Exportar datos" and "Atrás".

Figura 78. Características de la sección central ejemplo.

Elegidas las características, se hará clic en el botón *Exportar datos* para que la generación de la pieza se produzca. Aparecerá una ventana emergente como la de la Figura 20 en la que se podrá seleccionar dónde guardar la pieza y una vez que se guarde volverá a aparecer la ventana principal para que sigamos con el proceso.

Se van a definir consecuentemente dos clases de asientos, con sus correspondientes estantes, baño y galley.

Si se hace clic en el botón correspondiente al pasaje aparece la ventana que se muestra en la Figura 79, en la que se va a introducir dos clases para luego hacer clic en *Importar datos*:

Elección del número de clases

Número de clases

Importar datos

Atrás

Figura 79. Número de clases del ejemplo.

Al importar los datos, se elegirá la opción de uno o dos pasillos. En este ejemplo, se optará por una primera clase de un pasillo y una segunda clase de dos pasillos en la que se han elegido los siguientes datos para la primera y segunda clase respectivamente:

Pasaje de un pasillo

Offset delantero mm

Número de filas

Número de asientos por fila

Altura del asiento mm

Anchura del asiento mm

Longitud del asiento mm

Pitch mm

Separación entre asientos mm

Anchura del pasillo mm

Exportar datos

Atrás

Figura 80. Características de la primera clase.

Pasaje de dos pasillos ×

Offset delantero	<input type="text" value="2200"/>	Anchura del asiento	<input type="text" value="500"/>
Número de filas	<input type="text" value="4"/>	Longitud del asiento	<input type="text" value="500"/>
Número de asientos por fila centrales	<input type="text" value="3"/>	Pitch	<input type="text" value="100"/>
Número de asientos por fila extremos	<input type="text" value="2"/>	Separación entre asientos	<input type="text" value="50"/>
Altura del asiento	<input type="text" value="500"/>	Anchura de los pasillos	<input type="text" value="200"/>

Figura 81. Características de la segunda clase.

Conviene recordar que el proceso de guardado de las piezas se hará como el de la Figura 21 y que el nombre que se dará a cada una de las clases queda prefijado como Clase1, Clase2, etc. Una vez la ventana principal vuelva a salir, se acometerá la generación de los estantes de cada una de las clases, tomando especial atención en la posición que se le da para que el *product* final quede configurado de forma coherente. Al ser el procedimiento a seguir el mismo que para los asientos, se mostrarán únicamente las características de la primera y segunda clase respectivamente.

Estantes en distribución de un pasillo ×

Offset delantero	<input type="text" value="0"/>	mm
Porcentaje de la altura del asiento por encima del mismo	<input type="text" value="30"/>	%

Figura 82. Estantes de la primera clase.

Estantes en distribución de dos pasillos

Offset delantero mm

Porcentaje de la altura del asiento por encima del mismo %

Exportar datos

Atrás

Figura 83. Estantes de la segunda clase.

El nombre con el que se guardarán los nombres de los estantes será Carga1, Carga2, etc. Quedan por definir el baño y el galley.

Desde la ventana principal se hace clic en el botón correspondiente al baño. En la ventana que aparece se introducirán las características que va a tener el mismo siendo coherente una vez más con la posición en la que se va a colocar y la anchura que va a tener. Se han elegido las siguientes:

Restricciones del baño

Offset delantero mm

Anchura del baño mm

Profundidad mm

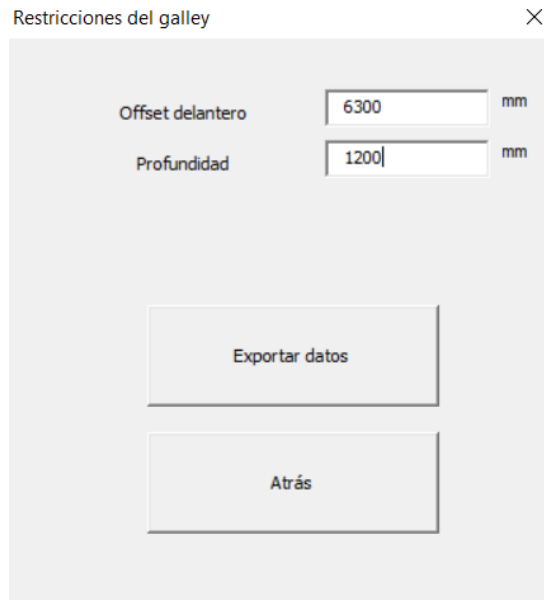
Exportar datos

Atrás

Figura 84. Características del baño.

El procedimiento de guardado de esta pieza vuelve a ser el mismo que el de la sección central.

Por último, se generará el galley accediendo a su formulario a través de su botón de la ventana principal. Las características que se han elegido para él son las siguientes:



Restricciones del galley ×

Offset delantero mm

Profundidad mm

Exportar datos

Atrás

Figura 85. Características del galley.

Una vez hecho esto, volverá a aparecer la ventana principal. El siguiente paso será conformar el *product* final con las piezas que se han creado y analizar las posibles interferencias. La ventana que nos permite conformar el *product* es la siguiente:



Figura 86. Selección de piezas que conformen el *product*.

Si el proceso se realiza correctamente se exportará automáticamente el archivo *.xml*, en la ubicación la cual se hayan ido guardando las piezas y con él se podrá visualizar los posibles conflictos entre los *parts* y si es viable la disposición en la que se ha ido trabajando. El análisis además abrirá el archivo *.xml* en el navegador que se elija a través de una ventana que dará la posibilidad de elegir la aplicación con la que abrir el mismo. En el presente caso, se abrió automáticamente Internet Explorer. Por último se mostrarán unas imágenes tanto del *product* obtenido como de la ventana del navegador que muestra el análisis realizado.

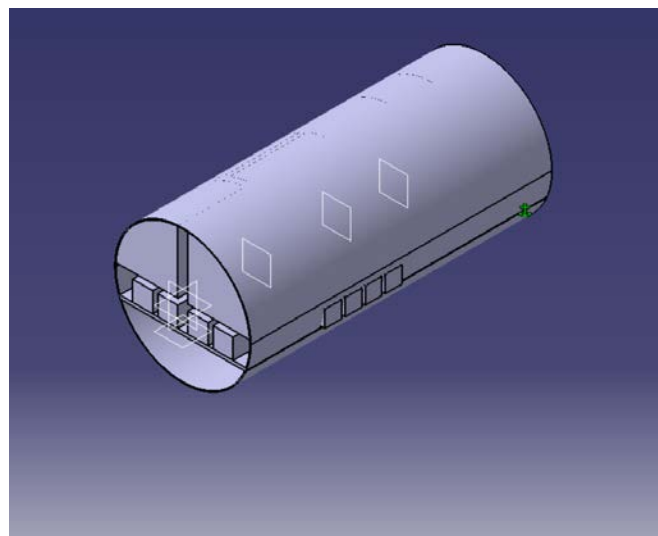


Figura 87. Ejemplo realizado I.

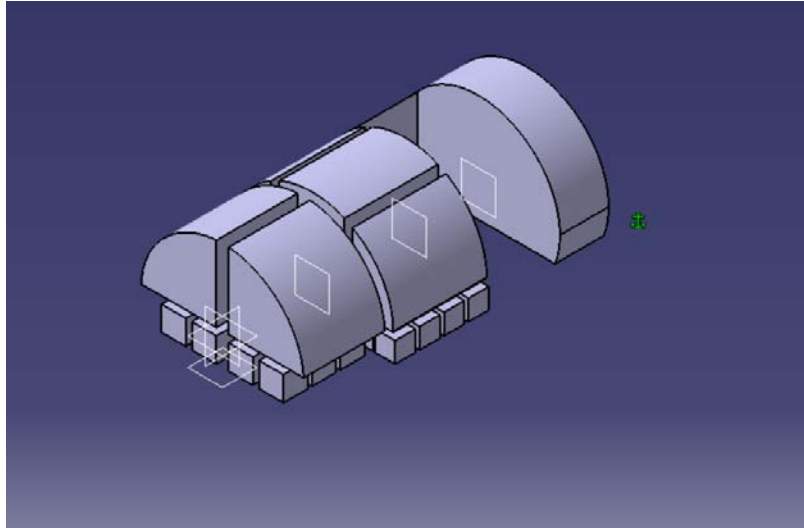


Figura 88. Ejemplo realizado II.

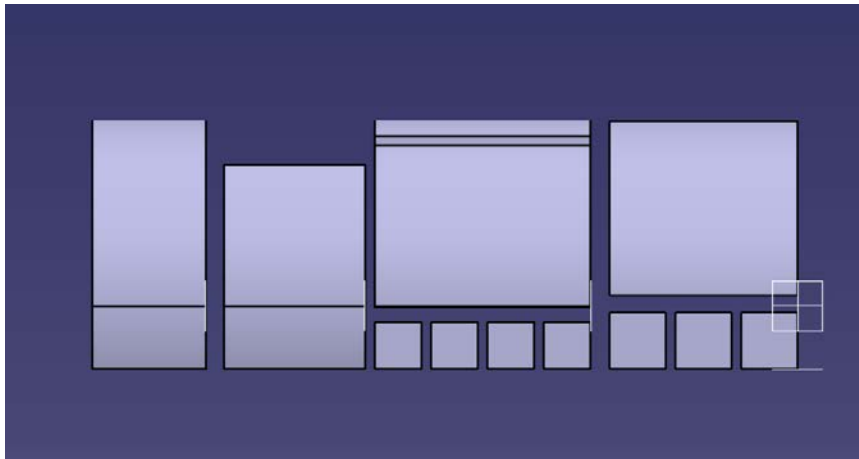


Figura 89. Vista lateral de ejemplo realizado.

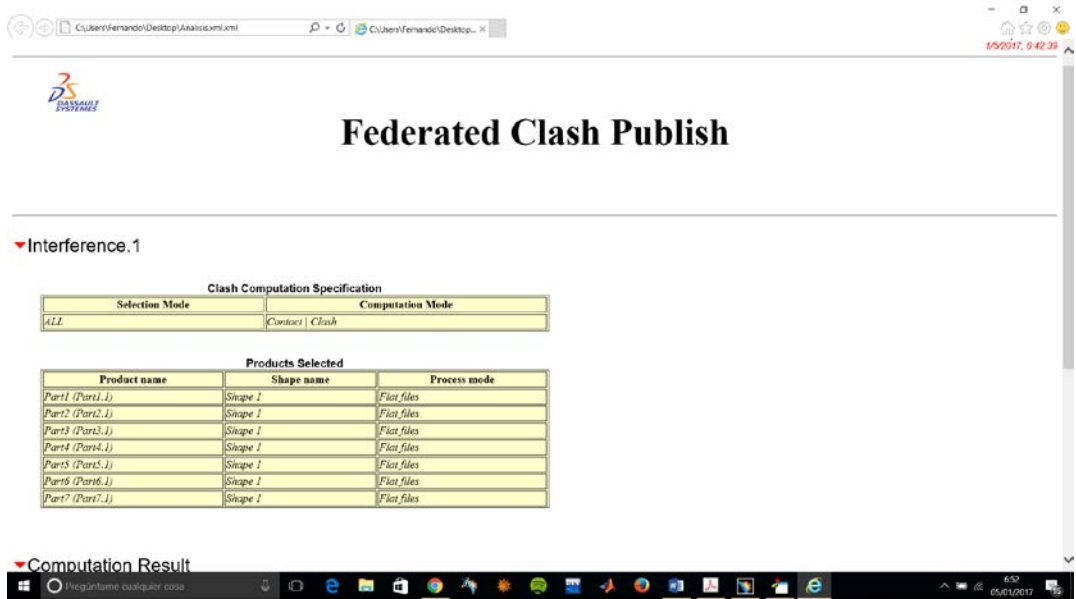


Figura 90. Análisis en el navegador web I.

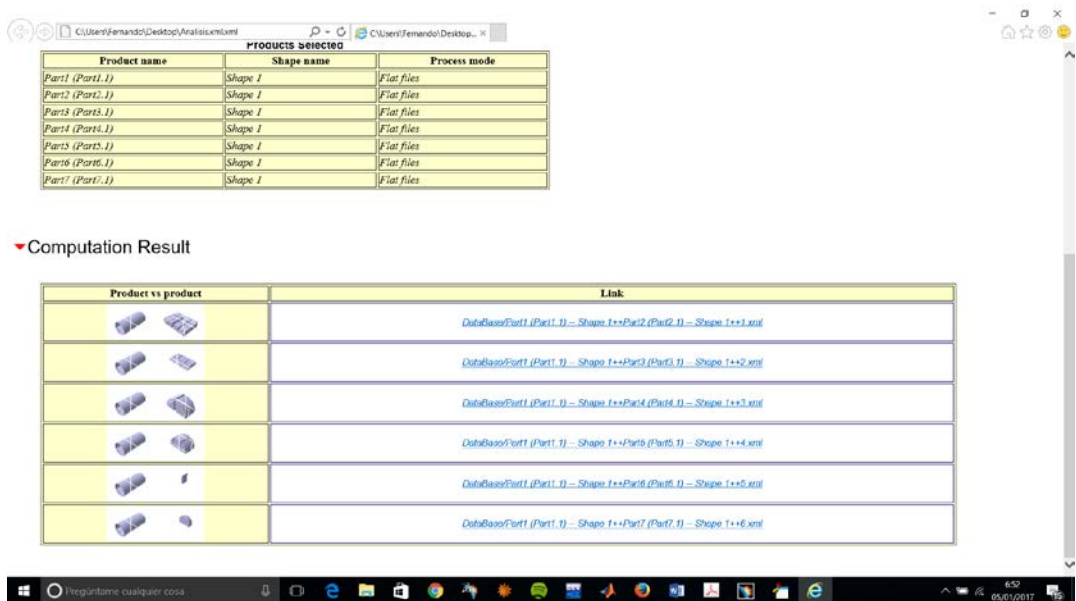


Figura 91. Análisis en el navegador web II.

9. Conclusiones

La forma de trabajar empleada en la herramienta académica presentada discurre en paralelo al proceso de ingeniería concurrente de la realidad. También es posible destacar la sencillez de uso de la misma el potencial de trabajo que puede proporcionar.

Se ha logrado que el departamento de Diseño no quede relegado a trabajar de forma independiente hasta las casi últimas etapas del diseño de forma que pueda integrarse con los departamentos de Actuaciones y Propulsión, Estabilidad, Aerodinámica y Estructuras.

Se ha obtenido una macro para CATIA V5 programada en VBA con la cual a partir de ciertos parámetros iniciales característicos de la geometría de diferentes piezas, la aplicación sea capaz de hacer análisis que nos evalúe en cierto modo cómo de viable es la idea que el usuario tiene en la cabeza.

La propia aplicación está concebida y así se refleja en su forma de proceder para que exista una interacción entre el usuario y ella, dotando al usuario de apoyo pero no de un trabajo automático.

Se entiende que éste es un primer paso de una herramienta que va a estar en desarrollo permanentemente y que será puesta a prueba por los propios alumnos.

10. Líneas futuras

La línea de trabajo futura debe ir encaminada en la mejora de la programación de la aplicación para que sea más robusta, sobre todo en la parte de los formularios y en la transferencia de datos entre unos y otros. También se podría ahondar en la elección de parámetros geométricos para la definición de cada una de las piezas, haciendo que el alcance de la herramienta sea mucho mayor. Incluso se podría dar la posibilidad de elección de varios pallets y/o contenedores que vayan a disponerse en la bodega.

Por otra parte, el estudio sobre la integración de más módulos a la aplicación tales como la generación de posibles sistemas embarcados, su distribución en la aeronave y las posibles interferencias que ello causaría o la generación automática y distribución de elementos de la cabina de los pilotos y la cola del avión junto con su correspondiente estudio de colisiones podría ser otra mejora.

También se podrían añadir elementos correspondientes a la fase detallada de diseño de aeronaves tales como el estudio del tren de aterrizaje, los requisitos de despegue, el dimensionamiento de los neumáticos así como la implementación de técnicas de optimización como las matrices de dimensionado o las gráficas de alfombra, estas últimas al menos no para el usuario sino como herramienta para que el profesor disponga de un sistema automatizado de optimización de parámetros sometidos a las restricciones del problema de la asignatura.

11. Bibliografía

- [1]. Dassault Systèmes Web Site. 20/11/2016
- [2]. Cristina Torrecillas. *Introducción a la programación Visual Basic en CATIA V5 Y V6* (no publicado).
- [3]. Emmett Ross. *VB Scripting for CATIA V5*. Segunda edición.
- [4]. Dieter Ziethen. *CATIA V5 Macro Programming with Visual Basic Script*.
- [5]. Eduardo Torrecilla Insagurbe. *El gran libro de CATIA*.
- [6]. Ajoy Kumar Kundu. *Aircraft Design*.
- [7]. Dr. Jan Roskam. *Airplane Design Part II*.
- [8]. Dr. Jan Roskam. *Airplane Design Part III*.

12. Anexo 1: Códigos de programación

```
.....  
.....  
' NOMBRE DEL TFG: DESARROLLO DE UNA HERRAMIENTA ACADÉMICA PARA LA GENERACIÓN Y  
DISTRIBUCIÓN DE INTERIORES ORIENTADA AL DISEÑO DE AERONAVES BASADA EN CATIA  
' AUTOR DEL TFG: FERNANDO ORTEGA MESAS  
' CONTACTO: fernandoortegamesas@gmail.com  
.....  
.....
```

- **VentanaPrincipal.**
Private Sub CommandButton1_Click()

```
Load SecciónCentral  
Unload Me  
SecciónCentral.Show
```

End Sub

```
Private Sub CommandButton2_Click()
```

```
Load EligeClases  
Unload Me  
EligeClases.Show
```

End Sub

```
Private Sub CommandButton3_Click()
```

```
Load Pasajero  
Unload Me  
Pasajero.Show
```

End Sub

```
Private Sub CommandButton4_Click()
```

```
Load Carga  
Unload Me  
Carga.Show
```

End Sub

```
Private Sub CommandButton5_Click()
```

```
Load Bodega  
Unload Me  
Bodega.Show
```

End Sub

```
Private Sub CommandButton6_Click()
```

```
Load Baño  
Unload Me  
Baño.Show
```

End Sub

```
Private Sub CommandButton7_Click()
```

```
Load Galley  
Unload Me  
Galley.Show
```

```

End Sub

Private Sub CommandButton8_Click()

Unload Me

End Sub

Private Sub CommandButton9_Click()

Load ConformaProducto
Unload Me
ConformaProducto.Show

End Sub

```

- **Sección Central.**
Private Sub CommandButton1_Click()

```

' DEFINICIÓN DE VARIABLES

```

```

Dim Mr As Double
Dim Mr_ As Double
Dim E As Double
Dim H_s As Double
Dim S As Double
Dim E_s As Double
Dim P As Double
Dim A As Double
Dim B As Double
Dim C As Double
Dim X As Double

```

```

Mr = TextBox1.Value
Mr_ = TextBox2.Value
E = TextBox3.Value
H_s = TextBox4.Value
E_s = TextBox5.Value
P = TextBox6.Value

```

```

S = Mr_ - H_s
A = S / Mr_
B = A ^ 2
C = 1 - B
X = Mr * Sqr(C)

```

```

' TRASPASO DE DATOS ENTRE FORMULARIOS

```

```

Load Pasaje1
Pasaje1.TextBox10.Value = TextBox2.Value
Pasaje1.TextBox11.Value = TextBox4.Value

```

```

Load Pasaje2
Pasaje2.TextBox11.Value = TextBox2.Value
Pasaje2.TextBox12.Value = TextBox4.Value

```

```

Load Pasajero
Pasajero.TextBox4.Value = TextBox2.Value
Pasajero.TextBox5.Value = TextBox4.Value

```

```

Load Carga1
Carga1.TextBox3.Value = TextBox1.Value
Carga1.TextBox4.Value = TextBox2.Value

```


Carga1.TextBox5.Value = TextBox4.Value

Load Carga2

Carga2.TextBox3.Value = TextBox1.Value

Carga2.TextBox4.Value = TextBox2.Value

Carga2.TextBox5.Value = TextBox4.Value

Load Mercancía

Mercancía.TextBox5.Value = TextBox2.Value

Mercancía.TextBox6.Value = TextBox4.Value

Load Bodega

Bodega.TextBox6.Value = TextBox2.Value

Bodega.TextBox7.Value = TextBox4.Value

Bodega.TextBox8.Value = TextBox5.Value

Load Baño

Baño.TextBox4.Value = TextBox1.Value

Baño.TextBox5.Value = TextBox2.Value

Baño.TextBox6.Value = TextBox4.Value

Load Galley

Galley.TextBox3.Value = TextBox1.Value

Galley.TextBox4.Value = TextBox2.Value

Galley.TextBox5.Value = TextBox4.Value

' DEFINICIÓN DEL ENTORNO DE TRABAJO

Dim documents1 As Documents

Dim partDocument1 As PartDocument

Dim specsAndGeomWindow1 As SpecsAndGeomWindow

Dim part1 As Part

Dim bodies1 As Bodies

Dim body1 As Body

Dim sketches1 As Sketches

Dim originElements1 As OriginElements

Dim reference1 As Reference

Dim sketch1 As Sketch

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 documents1 = CATIA.Documents

Set partDocument1 = documents1.Add("Part")

Set specsAndGeomWindow1 = CATIA.ActiveWindow

Set part1 = partDocument1.Part

Set bodies1 = part1.Bodies

Set body1 = bodies1.Item("PartBody")

Set sketches1 = body1.Sketches

Set originElements1 = part1.OriginElements

Set reference1 = originElements1.PlaneYZ

Set sketch1 = sketches1.Add(reference1)

Set sketch1Variant = sketch1

```

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch1.Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

.....

' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

.....

' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

.....

' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

.....

' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

.....

' PRIMERA ELIPSE
.....

Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(0#, 0#, Mr, Mr_, Mr, Mr_)

Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")

ellipse2D1.CenterPoint = point2D1

ellipse2D1.ReportName = 3

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference2 As Reference
Set reference2 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference2)

constraint1.Mode = catCstModeDrivingDimension

```

```

Dim length1 As Length
Set length1 = constraint1.Dimension

length1.Value = Mr

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference3)

constraint2.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint2.Dimension

length2.Value = Mr_

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(ellipse2D1)

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeAngle, reference4, reference5)

constraint3.Mode = catCstModeDrivingDimension

constraint3.AngleSector = catCstAngleSector0

Dim angle1 As Angle
Set angle1 = constraint3.Dimension

angle1.Value = 0#

.....
' SEGUNDA ELIPSE
.....

Dim ellipse2D2 As Ellipse2D
Set ellipse2D2 = factory2D1.CreateClosedEllipse(0#, 0#, Mr + E, Mr_ + E, Mr + E, Mr_ + E)

ellipse2D2.CenterPoint = point2D1

ellipse2D2.ReportName = 4

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(ellipse2D2)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference6)

constraint4.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint4.Dimension

length3.Value = Mr + E

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(ellipse2D2)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference7)

constraint5.Mode = catCstModeDrivingDimension

```

```

Dim length4 As Length
Set length4 = constraint5.Dimension

length4.Value = Mr_ + E

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(ellipse2D2)

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D1)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeAngle, reference8, reference9)

constraint6.Mode = catCstModeDrivingDimension

constraint6.AngleSector = catCstAngleSector0

Dim angle2 As Angle
Set angle2 = constraint6.Dimension

angle2.Value = 0#

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

.....
' PAD SECCIÓN CENTRAL
.....

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, P)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length5 As Length
Set length5 = limit1.Dimension

length5.Value = P

part1.UpdateObject pad1
part1.Update

.....
' SUELO SECCIÓN CENTRAL
.....

Dim sketch2 As Sketch
Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = 0#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#

```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
Set sketch2 = sketches1.Add(reference1)
Set sketch2Variant = sketch2
```

```
sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2
```

```
.....
' CONJUNTO DE HERRAMIENTAS 2D
.....
```

```
Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()
```

```
.....
' ELEMENTOS GEOMÉTRICOS
.....
```

```
Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements
```

```
.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....
```

```
Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
```

```
.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....
```

```
Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")
```

```
line2D3.ReportName = 1
```

```
Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")
```

```
line2D4.ReportName = 2
```

```
Dim point2D2 As Point2D
Set point2D2 = factory2D2.CreatePoint(-X, -(S + E_s))
```

```
point2D2.ReportName = 3
```

```
Dim point2D3 As Point2D
Set point2D3 = factory2D2.CreatePoint(-X, -S)
```

```
point2D3.ReportName = 4
```

```
Dim line2D5 As Line2D
Set line2D5 = factory2D2.CreateLine(-X, -(S + E_s), -X, -S)
```

```
line2D5.ReportName = 5
```

```
line2D5.StartPoint = point2D2
```

```
line2D5.EndPoint = point2D3
```

```
Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints
```

```

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D5)

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D4)

Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeVerticality, reference10, reference11)

constraint7.Mode = catCstModeDrivingDimension

Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(X, -S)

point2D4.ReportName = 6

Dim line2D6 As Line2D
Set line2D6 = factory2D2.CreateLine(-X, -S, X, -S)

line2D6.ReportName = 7

line2D6.StartPoint = point2D3

line2D6.EndPoint = point2D4

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D6)

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D3)

Dim constraint8 As Constraint
Set constraint8 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference12, reference13)

constraint8.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D
Set point2D5 = factory2D2.CreatePoint(X, -(S + E_s))

point2D5.ReportName = 8

Dim line2D7 As Line2D
Set line2D7 = factory2D2.CreateLine(X, -S, X, -(S + E_s))

line2D7.ReportName = 9

line2D7.StartPoint = point2D4

line2D7.EndPoint = point2D5

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D7)

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D4)

Dim constraint9 As Constraint
Set constraint9 = constraints2.AddBiEltCst(catCstTypeVerticality, reference14, reference15)

constraint9.Mode = catCstModeDrivingDimension

Dim line2D8 As Line2D
Set line2D8 = factory2D2.CreateLine(X, -(S + E_s), -X, -(S + E_s))

line2D8.ReportName = 10

```

```

line2D8.StartPoint = point2D5

line2D8.EndPoint = point2D2

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D8)

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D3)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference16, reference17)

constraint10.Mode = catCstModeDrivingDimension

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D3)

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D6)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeDistance, reference18, reference19)

constraint11.Mode = catCstModeDrivingDimension

Dim length6 As Length
Set length6 = constraint11.Dimension

length6.Value = S

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D6)

Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D8)

Dim constraint12 As Constraint
Set constraint12 = constraints2.AddBiEltCst(catCstTypeDistance, reference20, reference21)

constraint12.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint12.Dimension

length7.Value = E_s

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D4)

Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D7)

Dim constraint13 As Constraint
Set constraint13 = constraints2.AddBiEltCst(catCstTypeDistance, reference22, reference23)

constraint13.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint13.Dimension

length8.Value = X

Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D7)

Dim reference25 As Reference

```

```

Set reference25 = part1.CreateReferenceFromObject(line2D5)

Dim constraint14 As Constraint
Set constraint14 = constraints2.AddBiEltCst(catCstTypeDistance, reference24, reference25)

constraint14.Mode = catCstModeDrivingDimension

Dim length9 As Length
Set length9 = constraint14.Dimension

length9.Value = 2 * X

sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

' PAD SUELO SECCIÓN CENTRAL
'
Dim pad2 As Pad
Set pad2 = shapeFactory1.AddNewPad(sketch2, P)

pad2.DirectionOrientation = catInverseOrientation

part1.UpdateObject pad2
part1.Update

' FIT ALL IN
'
Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

' GUARDADO DEL PART
'
Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath
Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)
Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing

partDocument1.SaveAs objPath & "\\SeccionCentral.CATPart"

' TRASPASO DE LA RUTA DE GUARDADO
'
Load ConformaProducto

```



```
ConformaProducto.TextBox1.Text = objPath
```

```
.....  
' CERRAMOS EL FORMULARIO CORRESPONDIENTE  
.....
```

```
SecciónCentral.Hide  
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load VentanaPrincipal  
Unload Me  
VentanaPrincipal.Show
```

```
End Sub
```

- **EligeClases.**

```
Private Sub CommandButton1_Click()
```

```
Dim Nc As Double
```

```
Nc = TextBox1.Value
```

```
For i = 1 To Nc
```

```
    EligePasillo.Show
```

```
Next i
```

```
EligeClases.Hide  
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load VentanaPrincipal  
Unload Me  
VentanaPrincipal.Show
```

```
End Sub
```

- **EligePasillo.**

```
Private Sub CommandButton1_Click()
```

```
Load Pasaje1  
Unload Me  
Pasaje1.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load Pasaje2  
Unload Me  
Pasaje2.Show
```

```
End Sub
```

```
Private Sub CommandButton3_Click()
```

```
Load EligeClases
```

Unload Me
EligeClases.Show

End Sub

- **Pasaje1.**

Private Sub CommandButton1_Click()

.....

' DEFINICIÓN DE VARIABLES

.....

Dim Od As Double
Dim Nf As Double
Dim NAf As Double
Dim ALa As Double
Dim Aa As Double
Dim La As Double
Dim P As Double
Dim SEa As Double
Dim Ap As Double
Dim Mr_ As Double
Dim H_s As Double
Dim S As Double

Od = TextBox1.Value
Nf = TextBox2.Value
NAf = TextBox3.Value
ALa = TextBox4.Value
Aa = TextBox5.Value
La = TextBox6.Value
P = TextBox7.Value
SEa = TextBox8.Value
Ap = TextBox9.Value
Mr_ = TextBox10.Value
H_s = TextBox11.Value

S = Mr_ - H_s

.....

' TRASPASO DE DATOS ENTRE FORMULARIOS

.....

Load Carga1

Carga1.TextBox6.Value = TextBox2.Value
Carga1.TextBox7.Value = TextBox4.Value
Carga1.TextBox8.Value = TextBox6.Value
Carga1.TextBox9.Value = TextBox7.Value
Carga1.TextBox10.Value = TextBox9.Value

.....

' DEFINICIÓN DEL ENTORNO DE TRABAJO

.....

Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body

Set documents1 = CATIA.Documents

```

Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, S#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")

```

```
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
```

```

Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = -S#
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#

```

```

Set sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1

```

```

'.....
' SISTEMA DE EJES ABSOLUTOS
'.....

```

```

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

```

```

'.....
' CONJUNTO DE HERRAMIENTAS 2D
'.....

```

```

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

```

```

'.....
' ELEMENTOS GEOMÉTRICOS
'.....

```

```

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

```

```

'.....
' SISTEMA DE EJES DENTRO DEL SKETCH
'.....

```

```

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

```

```

'.....
' DIRECCIONES HORIZONTAL Y VERTICAL
'.....

```

```

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

' PASAJE DE UN PASILLO
' PASAJE DE UN PASILLO

Dim point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(-(Od + La), -Ap / 2)

point2D1.ReportName = 3

Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(-Od, -Ap / 2)

point2D2.ReportName = 4

Dim line2D3 As Line2D
Set line2D3 = factory2D1.CreateLine(-(Od + La), -Ap / 2, -Od, -Ap / 2)

line2D3.ReportName = 5

line2D3.StartPoint = point2D1

line2D3.EndPoint = point2D2

Dim point2D3 As Point2D
Set point2D3 = factory2D1.CreatePoint(-Od, -((Ap / 2) + Aa))

point2D3.ReportName = 6

Dim line2D4 As Line2D
Set line2D4 = factory2D1.CreateLine(-Od, -Ap / 2, -Od, -((Ap / 2) + Aa))

line2D4.ReportName = 7

line2D4.EndPoint = point2D2

line2D4.StartPoint = point2D3

Dim point2D4 As Point2D
Set point2D4 = factory2D1.CreatePoint(-(Od + La), -((Ap / 2) + Aa))

point2D4.ReportName = 8

Dim line2D5 As Line2D
Set line2D5 = factory2D1.CreateLine(-Od, -((Ap / 2) + Aa), -(Od + La), -((Ap / 2) + Aa))

line2D5.ReportName = 9

line2D5.StartPoint = point2D3

line2D5.EndPoint = point2D4

Dim line2D6 As Line2D
Set line2D6 = factory2D1.CreateLine(-(Od + La), -((Ap / 2) + Aa), -(Od + La), -Ap / 2)

line2D6.ReportName = 10

```

```

line2D6.EndPoint = point2D4

line2D6.StartPoint = point2D1

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D3)

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference3, reference4)

constraint1.Mode = catCstModeDrivingDimension

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D5)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference5, reference6)

constraint2.Mode = catCstModeDrivingDimension

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D4)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint3.Mode = catCstModeDrivingDimension

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D2)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeVerticality, reference9, reference10)

constraint4.Mode = catCstModeDrivingDimension

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D1)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeDistance, reference11, reference12)

constraint5.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint5.Dimension

length1.Value = Ap / 2

```

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D3)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D5)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeDistance, reference13, reference14)

constraint6.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint6.Dimension

length2.Value = Aa

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D2)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D4)

Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint7.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint7.Dimension

length3.Value = Od

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D4)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints1.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = La

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

.....
' PAD PASAJE DE UN PASILLO
.....

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, ALa)

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length5 As Length

```

Set length5 = limit1.Dimension

length5.Value = ALa

part1.Update

' RECTANGULAR PATTERN

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim rectPattern1 As RectPattern
Set rectPattern1 = shapeFactory1.AddNewRectPattern(Nothing, Nf, NAF, La + P, Aa + SEa, Nf, NAF, reference19,
reference20, True, True, 0#)

rectPattern1.SetFirstDirection reference19
rectPattern1.SetSecondDirection reference20

rectPattern1.FirstRectangularPatternParameters = catInstancesandSpacing
rectPattern1.SecondRectangularPatternParameters = catInstancesandSpacing

Dim linearRepartition1 As LinearRepartition
Set linearRepartition1 = rectPattern1.FirstDirectionRepartition

Dim length6 As Length
Set length6 = linearRepartition1.Spacing

length6.Value = La + P

Dim linearRepartition2 As LinearRepartition
Set linearRepartition2 = rectPattern1.FirstDirectionRepartition

Dim intParam1 As IntParam
Set intParam1 = linearRepartition2.InstancesCount

intParam1.Value = Nf

Dim linearRepartition3 As LinearRepartition
Set linearRepartition3 = rectPattern1.SecondDirectionRepartition

Dim intParam2 As IntParam
Set intParam2 = linearRepartition3.InstancesCount

intParam2.Value = NAF

Dim linearRepartition4 As LinearRepartition
Set linearRepartition4 = rectPattern1.SecondDirectionRepartition

Dim length7 As Length
Set length7 = linearRepartition4.Spacing

length7.Value = Aa + SEa

part1.Update

' MIRROR

Dim hybridShapePlaneExplicit2 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit2 = originElements1.PlaneZX

```

```
Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit2)
```

```
Dim mirror1 As Mirror
Set mirror1 = shapeFactory1.AddNewMirror(reference21)
```

```
part1.Update
```

```
.....
' FIT ALL IN
.....
```

```
Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow
```

```
Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer
```

```
viewer3D1.Reframe
```

```
Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D
```

```
.....
' GUARDADO DEL PART
.....
```

```
Dim RutaPart As String
Dim MensajePrecaución As String
RutaPart = CATIA.FileSelectionBox("SaveAs", "*.CATPart", 1)
```

```
'Posible Cancelación
If RutaPart = "" Then
```

```
    MensajePrecaución = "La operación de guardado fue cancelada"
    MsgBox MensajePrecaución, 48, "Part no guardado"
    Exit Sub
```

```
End If
```

```
'Guardado
partDocument1.SaveAs RutaPart
```

```
.....
' CERRAMOS EL FORMULARIO CORRESPONDIENTE
.....
```

```
Pasaje1.Hide
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load EligePasillo
Unload Me
EligePasillo.Show
```

```
End Sub
```

- **Pasaje2.**

```
Private Sub CommandButton1_Click()
```



```
.....
' DEFINICIÓN DE VARIABLES
.....
```

```
Dim Od1 As Double
Dim Nf1 As Double
Dim NAfc1 As Double
Dim NAfe1 As Double
Dim ALa1 As Double
Dim Aa1 As Double
Dim La1 As Double
Dim P1 As Double
Dim SEa1 As Double
Dim Ap1 As Double
Dim Mr_ As Double
Dim H_s As Double
Dim S As Double
```

```
Od1 = TextBox1.Value
Nf1 = TextBox2.Value
NAfc1 = TextBox3.Value
NAfe1 = TextBox4.Value
ALa1 = TextBox5.Value
Aa1 = TextBox6.Value
La1 = TextBox7.Value
P1 = TextBox8.Value
SEa1 = TextBox9.Value
Ap1 = TextBox10.Value
Mr_ = TextBox11.Value
H_s = TextBox12.Value
```

```
S = Mr_ - H_s
Lt = NAfc1 * Aa1 + (NAfc1 - 1) * SEa1
```

```
.....
' TRASPASO DE DATOS ENTRE FORMULARIOS
.....
```

```
Load Carga2
Carga2.TextBox6.Value = TextBox2.Value
Carga2.TextBox7.Value = TextBox3.Value
Carga2.TextBox8.Value = TextBox5.Value
Carga2.TextBox9.Value = TextBox6.Value
Carga2.TextBox10.Value = TextBox7.Value
Carga2.TextBox11.Value = TextBox8.Value
Carga2.TextBox12.Value = TextBox9.Value
Carga2.TextBox13.Value = TextBox10.Value
```

```
.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....
```

```
Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body
```

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
```

```
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, S#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

```
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
```

```
Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = -S#
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#
```

```
Set sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
```

```
.....
' CONJUNTO DE HERRAMIENTAS 2D
.....
```

```
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
```

```
.....
' ELEMENTOS GEOMÉTRICOS
.....
```

```
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
```

```
.....
' SISTEMAS DE EJES DENTRO DEL SKETCH
.....
```

```
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
```

```
.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....
```

```
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
```

```

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

' ASIENTOS CENTRALES PASAJE DE DOS PASILLOS
' ASIENTOS CENTRALES PASAJE DE DOS PASILLOS

Dim point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(-(Od1 + La1), -Lt / 2)

point2D1.ReportName = 3

Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(-(Od1 + La1), -((Lt / 2) - Aa1))

point2D2.ReportName = 4

Dim line2D3 As Line2D
Set line2D3 = factory2D1.CreateLine(-(Od1 + La1), -Lt / 2, -(Od1 + La1), -((Lt / 2) - Aa1))

line2D3.ReportName = 5

line2D3.StartPoint = point2D1

line2D3.EndPoint = point2D2

Dim point2D3 As Point2D
Set point2D3 = factory2D1.CreatePoint(-Od1, -((Lt / 2) - Aa1))

point2D3.ReportName = 6

Dim line2D4 As Line2D
Set line2D4 = factory2D1.CreateLine(-(Od1 + La1), -((Lt / 2) - Aa1), -Od1, -((Lt / 2) - Aa1))

line2D4.ReportName = 7

line2D4.EndPoint = point2D2

line2D4.StartPoint = point2D3

Dim point2D4 As Point2D
Set point2D4 = factory2D1.CreatePoint(-Od1, -Lt / 2)

point2D4.ReportName = 8

Dim line2D5 As Line2D
Set line2D5 = factory2D1.CreateLine(-Od1, -((Lt / 2) - Aa1), -Od1, -Lt / 2)

line2D5.ReportName = 9

line2D5.StartPoint = point2D3

line2D5.EndPoint = point2D4

Dim line2D6 As Line2D
Set line2D6 = factory2D1.CreateLine(-Od1, -Lt / 2, -(Od1 + La1), -Lt / 2)

line2D6.ReportName = 10

line2D6.EndPoint = point2D4

line2D6.StartPoint = point2D1

```

```

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D3)

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference3, reference4)

constraint1.Mode = catCstModeDrivingDimension

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D5)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference5, reference6)

constraint2.Mode = catCstModeDrivingDimension

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D4)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint3.Mode = catCstModeDrivingDimension

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D2)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeVerticality, reference9, reference10)

constraint4.Mode = catCstModeDrivingDimension

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D1)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeDistance, reference11, reference12)

constraint5.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint5.Dimension

length1.Value = (Lt / 2) - Aa1

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D3)

```

```

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D5)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeDistance, reference13, reference14)

constraint6.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint6.Dimension

length2.Value = Aa1

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D2)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D4)

Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint7.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint7.Dimension

length3.Value = Od1

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D4)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints1.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = La1

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

' PAD ASIENTOS CENTRALES PASAJE DE DOS PASILLOS

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, ALa1)

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length5 As Length
Set length5 = limit1.Dimension

length5.Value = ALa1

```

part1.UpdateObject pad1
part1.Update

.....
' RECTANGULAR PATTERN ASIENTOS CENTRALES
.....

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim rectPattern1 As RectPattern
Set rectPattern1 = shapeFactory1.AddNewRectPattern(Nothing, Nf1, NAfc1, La1 + P1, Aa1 + SEa1, Nf1, NAfc1, reference19, reference20, True, False, 0#)

rectPattern1.SetFirstDirection reference19
rectPattern1.SetSecondDirection reference20

rectPattern1.FirstRectangularPatternParameters = catInstancesandSpacing
rectPattern1.SecondRectangularPatternParameters = catInstancesandSpacing

Dim linearRepartition1 As LinearRepartition
Set linearRepartition1 = rectPattern1.FirstDirectionRepartition

Dim intParam1 As IntParam
Set intParam1 = linearRepartition1.InstancesCount

intParam1.Value = Nf1

Dim linearRepartition2 As LinearRepartition
Set linearRepartition2 = rectPattern1.FirstDirectionRepartition

Dim length6 As Length
Set length6 = linearRepartition2.Spacing

length6.Value = La1 + P1

Dim linearRepartition3 As LinearRepartition
Set linearRepartition3 = rectPattern1.SecondDirectionRepartition

Dim intParam2 As IntParam
Set intParam2 = linearRepartition3.InstancesCount

intParam2.Value = NAfc1

Dim linearRepartition4 As LinearRepartition
Set linearRepartition4 = rectPattern1.SecondDirectionRepartition

Dim length7 As Length
Set length7 = linearRepartition4.Spacing

length7.Value = Aa1 + SEa1

part1.Update

.....
' INSERCIÓN DE UN NUEVO BODY
.....

Dim body2 As Body
Set body2 = bodies1.Add()

part1.Update

```
Dim sketches2 As Sketches
Set sketches2 = body2.Sketches
```

```
Dim sketch2 As Sketch
Set sketch2 = sketches2.Add(reference2)
```

```
Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = 0#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = -S#
arrayOfVariantOfDouble2(3) = 1#
arrayOfVariantOfDouble2(4) = 0#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 1#
arrayOfVariantOfDouble2(8) = 0#
Set sketch2Variant = sketch2
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2
```

```
.....
' CONJUNTO DE HERRAMIENTAS 2D
.....
```

```
Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()
```

```
.....
' ELEMENTOS GEOMÉTRICOS
.....
```

```
Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements
```

```
.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....
```

```
Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
```

```
.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....
```

```
Dim line2D7 As Line2D
Set line2D7 = axis2D2.GetItem("HDirection")
```

```
line2D7.ReportName = 1
```

```
Dim line2D8 As Line2D
Set line2D8 = axis2D2.GetItem("VDirection")
```

```
line2D8.ReportName = 2
```

```
.....
' ASIENTOS EXTREMOS PASAJE DE DOS PASILLOS
.....
```

```
Dim point2D5 As Point2D
Set point2D5 = factory2D2.CreatePoint(-(Od1 + La1), -(Lt / 2) + Ap1 + Aa1))
```

```

point2D5.ReportName = 3

Dim point2D6 As Point2D
Set point2D6 = factory2D2.CreatePoint(-(Od1 + La1), -((Lt / 2) + Ap1))

point2D6.ReportName = 4

Dim line2D9 As Line2D
Set line2D9 = factory2D2.CreateLine(-(Od1 + La1), -((Lt / 2) + Ap1 + Aa1), -(Od1 + La1), -((Lt / 2) + Ap1))

line2D9.ReportName = 5

line2D9.StartPoint = point2D5

line2D9.EndPoint = point2D6

Dim point2D7 As Point2D
Set point2D7 = factory2D2.CreatePoint(-Od1, -((Lt / 2) + Ap1))

point2D7.ReportName = 6

Dim line2D10 As Line2D
Set line2D10 = factory2D2.CreateLine(-(Od1 + La1), -((Lt / 2) + Ap1), -Od1, -((Lt / 2) + Ap1))

line2D10.ReportName = 7

line2D10.EndPoint = point2D6

line2D10.StartPoint = point2D7

Dim point2D8 As Point2D
Set point2D8 = factory2D2.CreatePoint(-Od1, -((Lt / 2) + Ap1 + Aa1))

point2D8.ReportName = 8

Dim line2D11 As Line2D
Set line2D11 = factory2D2.CreateLine(-Od1, -((Lt / 2) + Ap1), -Od1, -((Lt / 2) + Ap1 + Aa1))

line2D11.ReportName = 9

line2D11.StartPoint = point2D7

line2D11.EndPoint = point2D8

Dim line2D12 As Line2D
Set line2D12 = factory2D2.CreateLine(-Od1, -((Lt / 2) + Ap1 + Aa1), -(Od1 + La1), -((Lt / 2) + Ap1 + Aa1))

line2D12.ReportName = 10

line2D12.EndPoint = point2D8

line2D12.StartPoint = point2D5

Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints

Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D9)

Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D7)

Dim constraint9 As Constraint
Set constraint9 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference23, reference24)

constraint9.Mode = catCstModeDrivingDimension

```


Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(line2D11)

Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D7)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference25, reference26)

constraint10.Mode = catCstModeDrivingDimension

Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D10)

Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D8)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeVerticality, reference27, reference28)

constraint11.Mode = catCstModeDrivingDimension

Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(line2D12)

Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(line2D8)

Dim constraint12 As Constraint
Set constraint12 = constraints2.AddBiEltCst(catCstTypeVerticality, reference29, reference30)

constraint12.Mode = catCstModeDrivingDimension

Dim reference31 As Reference
Set reference31 = part1.CreateReferenceFromObject(line2D7)

Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D9)

Dim constraint13 As Constraint
Set constraint13 = constraints2.AddBiEltCst(catCstTypeDistance, reference31, reference32)

constraint13.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint13.Dimension

length8.Value = (Lt / 2) + Ap1

Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D9)

Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromObject(line2D11)

Dim constraint14 As Constraint
Set constraint14 = constraints2.AddBiEltCst(catCstTypeDistance, reference33, reference34)

constraint14.Mode = catCstModeDrivingDimension

Dim length9 As Length
Set length9 = constraint14.Dimension

length9.Value = Aa1

Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(line2D8)

```

Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(line2D10)

Dim constraint15 As Constraint
Set constraint15 = constraints2.AddBiEltCst(catCstTypeDistance, reference35, reference36)

constraint15.Mode = catCstModeDrivingDimension

Dim length10 As Length
Set length10 = constraint15.Dimension

length10.Value = Od1

Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromObject(line2D10)

Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(line2D12)

Dim constraint16 As Constraint
Set constraint16 = constraints2.AddBiEltCst(catCstTypeDistance, reference37, reference38)

constraint16.Mode = catCstModeDrivingDimension

Dim length11 As Length
Set length11 = constraint16.Dimension

length11.Value = La1

sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

' PAD ASIENTOS EXTREMOS PASAJE DE DOS PASILLOS
'
Dim pad2 As Pad
Set pad2 = shapeFactory1.AddNewPad(sketch2, ALa1)

part1.Update

' RECTANGULAR PATTERN ASIENTOS EXTREMOS
'
Dim reference39 As Reference
Set reference39 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim reference40 As Reference
Set reference40 = part1.CreateReferenceFromObject(hybridShapePlaneOffset1)

Dim rectPattern2 As RectPattern
Set rectPattern2 = shapeFactory1.AddNewRectPattern(Nothing, Nf1, NAfe1, La1 + P1, Aa1 + SEa1, Nf1, NAfe1,
reference39, reference40, True, True, 0#)

rectPattern2.SetFirstDirection reference39
rectPattern2.SetSecondDirection reference40

rectPattern2.FirstRectangularPatternParameters = catInstancesandSpacing
rectPattern2.SecondRectangularPatternParameters = catInstancesandSpacing

Dim linearRepartition5 As LinearRepartition
Set linearRepartition5 = rectPattern2.FirstDirectionRepartition

```

```

Dim intParam3 As IntParam
Set intParam3 = linearRepartition5.InstancesCount

intParam3.Value = Nf1

Dim linearRepartition6 As LinearRepartition
Set linearRepartition6 = rectPattern2.FirstDirectionRepartition

Dim length12 As Length
Set length12 = linearRepartition6.Spacing

length12.Value = La1 + P1

Dim linearRepartition7 As LinearRepartition
Set linearRepartition7 = rectPattern2.SecondDirectionRepartition

Dim intParam4 As IntParam
Set intParam4 = linearRepartition7.InstancesCount

intParam4.Value = NAfe1

Dim linearRepartition8 As LinearRepartition
Set linearRepartition8 = rectPattern2.SecondDirectionRepartition

Dim length13 As Length
Set length13 = linearRepartition8.Spacing

length13.Value = Aa1 + SEa1

part1.Update

' MIRROR ASIENTOS EXTREMOS
' MIRROR ASIENTOS EXTREMOS

Dim hybridShapePlaneExplicit2 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit2 = originElements1.PlaneZX

Dim reference43 As Reference
Set reference43 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit2)

Dim mirror1 As Mirror
Set mirror1 = shapeFactory1.AddNewMirror(reference43)

part1.Update

' FIT ALL IN
' FIT ALL IN

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

' GUARDADO DEL PART
' GUARDADO DEL PART

Dim RutaPart As String
Dim MensajePrecaución As String

```

```
RutaPart = CATIA.FileSelectionBox("SaveAs", "*.CATPart", 1)
```

```
'Posible Cancelación
```

```
If RutaPart = "" Then
```

```
    MensajePrecaución = "La operación de guardado fue cancelada"
```

```
    MsgBox MensajePrecaución, 48, "Part no guardado"
```

```
    Exit Sub
```

```
End If
```

```
'Guardado
```

```
partDocument1.SaveAs RutaPart
```

```
.....  
' CERRAMOS EL FORMULARIO CORRESPONDIENTE  
.....
```

```
Pasaje2.Hide
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
    Load EligePasillo
```

```
    Unload Me
```

```
    EligePasillo.Show
```

```
End Sub
```

- **Pasajero.**

```
Private Sub CommandButton1_Click()
```

```
.....  
' DEFINICIÓN DE VARIABLES  
.....
```

```
Dim Al As Double
```

```
Dim An As Double
```

```
Dim L As Double
```

```
Dim Mr_ As Double
```

```
Dim H_s As Double
```

```
Dim S As Double
```

```
Al = TextBox1.Value
```

```
An = TextBox2.Value
```

```
L = TextBox3.Value
```

```
Mr_ = TextBox4.Value
```

```
H_s = TextBox5.Value
```

```
S = Mr_ - H_s
```

```
.....  
' DEFINICIÓN DEL ENTORNO DE TRABAJO  
.....
```

```
Dim documents1 As Documents
```

```
Dim partDocument1 As PartDocument
```

```
Dim part1 As Part
```

```
Dim hybridShapeFactory1 As HybridShapeFactory
```

```
Dim originElements1 As OriginElements
```

```
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
```

```
Dim reference1 As Reference
```

```
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
```

```

Dim bodies1 As Bodies
Dim body1 As Body

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, S#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")

body1.InsertHybridShape hybridShapePlaneOffset1
part1.InWorkObject = hybridShapePlaneOffset1

part1.Update

Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = -S#
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#

Set sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1

'.....
' SISTEMA DE EJES ABSOLUTOS
'.....

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

'.....
' CONJUNTO DE HERRAMIENTAS 2D
'.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

'.....
' ELEMENTOS GEOMÉTRICOS
'.....

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

'.....
' SISTEMA DE EJES DENTRO DEL SKETCH
'.....

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

```

```
.....  
' DIRECCIONES HORIZONTAL Y VERTICAL  
.....
```

```
Dim line2D1 As Line2D  
Set line2D1 = axis2D1.GetItem("HDirection")
```

```
line2D1.ReportName = 1
```

```
Dim line2D2 As Line2D  
Set line2D2 = axis2D1.GetItem("VDirection")
```

```
line2D2.ReportName = 2
```

```
.....  
' PASAJERO  
.....
```

```
Dim point2D1 As Point2D  
Set point2D1 = factory2D1.CreatePoint(-L, An / 2)
```

```
point2D1.ReportName = 3
```

```
Dim point2D2 As Point2D  
Set point2D2 = factory2D1.CreatePoint(0, An / 2)
```

```
point2D2.ReportName = 4
```

```
Dim line2D3 As Line2D  
Set line2D3 = factory2D1.CreateLine(-L, An / 2, 0, An / 2)
```

```
line2D3.ReportName = 5
```

```
line2D3.StartPoint = point2D1
```

```
line2D3.EndPoint = point2D2
```

```
Dim point2D3 As Point2D  
Set point2D3 = factory2D1.CreatePoint(-70#, -190#)
```

```
point2D3.ReportName = 6
```

```
Dim line2D4 As Line2D  
Set line2D4 = factory2D1.CreateLine(0, An / 2, 0, -An / 2)
```

```
line2D4.ReportName = 7
```

```
line2D4.EndPoint = point2D2
```

```
line2D4.StartPoint = point2D3
```

```
Dim point2D4 As Point2D  
Set point2D4 = factory2D1.CreatePoint(-L, -An / 2)
```

```
point2D4.ReportName = 8
```

```
Dim line2D5 As Line2D  
Set line2D5 = factory2D1.CreateLine(0, -An / 2, -L, -An / 2)
```

```
line2D5.ReportName = 9
```

```
line2D5.StartPoint = point2D3
```

```
line2D5.EndPoint = point2D4
```

```
Dim line2D6 As Line2D
```

```

Set line2D6 = factory2D1.CreateLine(-L, -An / 2, -L, An / 2)

line2D6.ReportName = 10

line2D6.EndPoint = point2D4

line2D6.StartPoint = point2D1

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D3)

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference3, reference4)

constraint1.Mode = catCstModeDrivingDimension

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D5)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference5, reference6)

constraint2.Mode = catCstModeDrivingDimension

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D4)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint3.Mode = catCstModeDrivingDimension

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D2)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeVerticality, reference9, reference10)

constraint4.Mode = catCstModeDrivingDimension

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D1)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeDistance, reference11, reference12)

constraint5.Mode = catCstModeDrivingDimension

Dim length1 As Length

```

```

Set length1 = constraint5.Dimension

length1.Value = An / 2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D3)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D5)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeDistance, reference13, reference14)

constraint6.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint6.Dimension

length2.Value = An

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D2)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D4)

Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint7.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint7.Dimension

length3.Value = 0

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D4)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints1.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = L

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

' PAD PASAJERO

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, A1)

```



```
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
```

```
Dim length5 As Length
Set length5 = limit1.Dimension
```

```
length5.Value = AI
```

```
part1.UpdateObject pad1
part1.Update
```

```
*****
' FIT ALL IN
*****
```

```
Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow
```

```
Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer
```

```
viewer3D1.Reframe
```

```
Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D
```

```
*****
' GUARDADO DEL PART
*****
```

```
Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath
Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)
Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing
```

```
partDocument1.SaveAs objPath & "\Pasajero.CATPart"
```

```
*****
' CERRAMOS EL FORMULARIO CORRESPONDIENTE
*****
```

```
Pasajero.Hide
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load VentanaPrincipal
Unload Me
VentanaPrincipal.Show
```

```
End Sub
```

- **Carga.**

```
Private Sub CommandButton1_Click()
```

Load EligeClases1
Unload Me
EligeClases1.Show

End Sub

Private Sub CommandButton2_Click()

Load Mercancía
Unload Me
Mercancía.Show

End Sub

Private Sub CommandButton3_Click()

Load VentanaPrincipal
Unload Me
VentanaPrincipal.Show

End Sub

- **EligeClases1.**

Private Sub CommandButton1_Click()

Dim Nc1 As Double

Nc1 = TextBox1.Value

For i = 1 To Nc1

 EligePasillo1.Show

Next i

EligeClases1.Hide
VentanaPrincipal.Show

End Sub

Private Sub CommandButton2_Click()

Load Carga
Unload Me
Carga.Show

End Sub

- **EligePasillo1.**

Private Sub CommandButton1_Click()

Load Carga1
Unload Me
Carga1.Show

End Sub

Private Sub CommandButton2_Click()

Load Carga2
Unload Me
Carga2.Show

End Sub

```
Private Sub CommandButton3_Click()
```

```
Load EligeClases1  
Unload Me  
EligeClases1.Show
```

```
End Sub
```

- **Carga1.**

```
Private Sub CommandButton1_Click()
```

```
.....
```

```
' DEFINICIÓN DE VARIABLES
```

```
.....
```

```
Dim Oe As Double  
Dim Po As Double  
Dim Mr As Double  
Dim Mr_ As Double  
Dim H_s As Double  
Dim S As Double  
Dim Nf As Double  
Dim ALa As Double  
Dim La As Double  
Dim P As Double  
Dim Ap As Double  
Dim Pe As Double
```

```
Oe = TextBox1.Value  
Po = TextBox2.Value  
Mr = TextBox3.Value  
Mr_ = TextBox4.Value  
H_s = TextBox5.Value  
Nf = TextBox6.Value  
ALa = TextBox7.Value  
La = TextBox8.Value  
P = TextBox9.Value  
Ap = TextBox10.Value
```

```
S = Mr_ - H_s  
Pe = Nf * La + (Nf - 1) * P
```

```
.....
```

```
' DEFINICIÓN DEL ENTORNO DE TRABAJO
```

```
.....
```

```
Dim documents1 As Documents  
Dim partDocument1 As PartDocument  
Dim part1 As Part  
Dim hybridShapeFactory1 As HybridShapeFactory  
Dim originElements1 As OriginElements  
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit  
Dim reference1 As Reference  
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset  
Dim bodies1 As Bodies  
Dim body1 As Body
```

```
Set documents1 = CATIA.Documents  
Set partDocument1 = documents1.Add("Part")  
Set part1 = partDocument1.Part  
Set hybridShapeFactory1 = part1.HybridShapeFactory  
Set originElements1 = part1.OriginElements  
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ  
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
```

```
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, Oe#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

```
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
```

```
Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = -Oe#
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 sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
```

```
.....
' CONJUNTO DE HERRAMIENTAS 2D
.....
```

```
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
```

```
.....
' ELEMENTOS GEOMÉTRICOS
.....
```

```
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
```

```
.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....
```

```
Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
```

```
.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....
```

```
Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")
```

```
line2D1.ReportName = 1
```

```

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

' ESTANTES EN UN PASILLO

Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(0#, 0#, Mr, Mr_, Mr, Mr_)

Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")

ellipse2D1.CenterPoint = point2D1

ellipse2D1.ReportName = 3

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference3)

constraint1.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint1.Dimension

length1.Value = Mr

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference4)

constraint2.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint2.Dimension

length2.Value = Mr_

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(ellipse2D1)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeAngle, reference5, reference6)

constraint3.Mode = catCstModeDrivingDimension

constraint3.AngleSector = catCstAngleSector0

Dim angle1 As Angle
Set angle1 = constraint3.Dimension

angle1.Value = 0#

sketch1.CloseEdition

```

```

part1.InWorkObject = sketch1
part1.Update

.....

' PAD ESTANTES EN UN PASILLO
.....

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, Pe)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length3 As Length
Set length3 = limit1.Dimension

length3.Value = Pe

part1.Update

.....

' RESTRICCIÓN EN ALTURA
.....

Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = -Oe#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2

.....

' SISTEMA DE EJES ABSOLUTOS
.....

sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2

.....

' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()

.....

' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements

.....

```

' SISTEMA DE EJES DENTRO DEL SKETCH

```
Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
```

' DIRECCIONES HORIZONTAL Y VERTICAL

```
Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")
```

line2D3.ReportName = 1

```
Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")
```

line2D4.ReportName = 2

' SKETCH RESTRICCIÓN EN ALTURA

```
Dim point2D2 As Point2D
Set point2D2 = factory2D2.CreatePoint(-Mr, -Mr_)
```

point2D2.ReportName = 3

```
Dim point2D3 As Point2D
Set point2D3 = factory2D2.CreatePoint(-Mr, (1 + (Po / 100)) * ALa - S)
```

point2D3.ReportName = 4

```
Dim line2D5 As Line2D
Set line2D5 = factory2D2.CreateLine(-Mr, -Mr_, -Mr, (1 + (Po / 100)) * ALa - S)
```

line2D5.ReportName = 5

line2D5.StartPoint = point2D2

line2D5.EndPoint = point2D3

```
Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints
```

```
Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D5)
```

```
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D4)
```

```
Dim constraint4 As Constraint
Set constraint4 = constraints2.AddBiEltCst(catCstTypeVerticality, reference7, reference8)
```

constraint4.Mode = catCstModeDrivingDimension

```
Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(Mr, (1 + (Po / 100)) * ALa - S)
```

point2D4.ReportName = 6

```
Dim line2D6 As Line2D
Set line2D6 = factory2D2.CreateLine(-Mr, (1 + (Po / 100)) * ALa - S, Mr, (1 + (Po / 100)) * ALa - S)
```

line2D6.ReportName = 7

```
line2D6.StartPoint = point2D3

line2D6.EndPoint = point2D4

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference9, reference10)

constraint5.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D
Set point2D5 = factory2D2.CreatePoint(Mr, -Mr_)

point2D5.ReportName = 8

Dim line2D7 As Line2D
Set line2D7 = factory2D2.CreateLine(Mr, (1 + (Po / 100)) * ALa - S, Mr, -Mr_)

line2D7.ReportName = 9

line2D7.StartPoint = point2D4

line2D7.EndPoint = point2D5

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D7)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D4)

Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeVerticality, reference11, reference12)

constraint6.Mode = catCstModeDrivingDimension

Dim line2D8 As Line2D
Set line2D8 = factory2D2.CreateLine(Mr, -Mr_, -Mr, -Mr_)

line2D8.ReportName = 10

line2D8.StartPoint = point2D5

line2D8.EndPoint = point2D2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D8)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)

Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference13, reference14)

constraint7.Mode = catCstModeDrivingDimension

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D3)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
```



```

Set constraint8 = constraints2.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = (1 + (Po / 100)) * ALa - S

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D8)

Dim constraint9 As Constraint
Set constraint9 = constraints2.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint9.Mode = catCstModeDrivingDimension

Dim length5 As Length
Set length5 = constraint9.Dimension

length5.Value = Mr_ + (1 + (Po / 100)) * ALa - S

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D4)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D7)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeDistance, reference19, reference20)

constraint10.Mode = catCstModeDrivingDimension

Dim length6 As Length
Set length6 = constraint10.Dimension

length6.Value = Mr

Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D7)

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D5)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeDistance, reference21, reference22)

constraint11.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint11.Dimension

length7.Value = 2 * Mr
sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

.....
' POCKET RESTRICCIÓN EN ALTURA
.....

Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch2, Pe)

```

part1.Update

.....
' RESTRICCIÓN PASILLO
.....

Dim sketch3 As Sketch
Set sketch3 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble3(8)
arrayOfVariantOfDouble3(0) = -0e#
arrayOfVariantOfDouble3(1) = 0#
arrayOfVariantOfDouble3(2) = 0#
arrayOfVariantOfDouble3(3) = 0#
arrayOfVariantOfDouble3(4) = 1#
arrayOfVariantOfDouble3(5) = 0#
arrayOfVariantOfDouble3(6) = 0#
arrayOfVariantOfDouble3(7) = 0#
arrayOfVariantOfDouble3(8) = 1#
Set sketch3Variant = sketch3

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3
part1.InWorkObject = sketch3

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D3 As Factory2D
Set factory2D3 = sketch3.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements3 As GeometricElements
Set geometricElements3 = sketch3.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D3 As Axis2D
Set axis2D3 = geometricElements3.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D9 As Line2D
Set line2D9 = axis2D3.GetItem("HDirection")

line2D9.ReportName = 1

Dim line2D10 As Line2D
Set line2D10 = axis2D3.GetItem("VDirection")

line2D10.ReportName = 2

.....
' SKETCH RESTRICCIÓN PASILLO
.....

```

Dim point2D6 As Point2D
Set point2D6 = factory2D3.CreatePoint(-Ap / 2, (1 + (Po / 100)) * ALa - S)

point2D6.ReportName = 3

Dim point2D7 As Point2D
Set point2D7 = factory2D3.CreatePoint(-Ap / 2, Mr_)

point2D7.ReportName = 4

Dim line2D11 As Line2D
Set line2D11 = factory2D3.CreateLine(-Ap / 2, (1 + (Po / 100)) * ALa - S, -Ap / 2, Mr_)

line2D11.ReportName = 5

line2D11.StartPoint = point2D6

line2D11.EndPoint = point2D7

Dim constraints3 As Constraints
Set constraints3 = sketch3.Constraints

Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D11)

Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D10)

Dim constraint12 As Constraint
Set constraint12 = constraints3.AddBiEltCst(catCstTypeVerticality, reference23, reference24)

constraint12.Mode = catCstModeDrivingDimension

Dim point2D8 As Point2D
Set point2D8 = factory2D3.CreatePoint(Ap / 2, Mr_)

point2D8.ReportName = 6

Dim line2D12 As Line2D
Set line2D12 = factory2D3.CreateLine(-Ap / 2, Mr_, Ap / 2, Mr_)

line2D12.ReportName = 7

line2D12.StartPoint = point2D7

line2D12.EndPoint = point2D8

Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(line2D12)

Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D9)

Dim constraint13 As Constraint
Set constraint13 = constraints3.AddBiEltCst(catCstTypeHorizontal, reference25, reference26)

constraint13.Mode = catCstModeDrivingDimension

Dim point2D9 As Point2D
Set point2D9 = factory2D3.CreatePoint(Ap / 2, (1 + (Po / 100)) * ALa - S)

point2D9.ReportName = 8

Dim line2D13 As Line2D
Set line2D13 = factory2D3.CreateLine(Ap / 2, Mr_, Ap / 2, (1 + (Po / 100)) * ALa - S)

```

```

line2D13.ReportName = 9

line2D13.StartPoint = point2D8

line2D13.EndPoint = point2D9

Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D13)

Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D10)

Dim constraint14 As Constraint
Set constraint14 = constraints3.AddBiEltCst(catCstTypeVerticality, reference27, reference28)

constraint14.Mode = catCstModeDrivingDimension

Dim line2D14 As Line2D
Set line2D14 = factory2D3.CreateLine(Ap / 2, (1 + (Po / 100)) * ALa - S, -Ap / 2, (1 + (Po / 100)) * ALa - S)

line2D14.ReportName = 10

line2D14.StartPoint = point2D9

line2D14.EndPoint = point2D6

Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(line2D14)

Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(line2D9)

Dim constraint15 As Constraint
Set constraint15 = constraints3.AddBiEltCst(catCstTypeHorizontalilty, reference29, reference30)

constraint15.Mode = catCstModeDrivingDimension

Dim reference31 As Reference
Set reference31 = part1.CreateReferenceFromObject(line2D9)

Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D14)

Dim constraint16 As Constraint
Set constraint16 = constraints3.AddBiEltCst(catCstTypeDistance, reference31, reference32)

constraint16.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint16.Dimension

length8.Value = (1 + (Po / 100)) * ALa - S

Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D14)

Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromObject(line2D12)

Dim constraint17 As Constraint
Set constraint17 = constraints3.AddBiEltCst(catCstTypeDistance, reference33, reference34)

constraint17.Mode = catCstModeDrivingDimension

Dim length9 As Length
Set length9 = constraint17.Dimension

```

```

length9.Value = Mr_ - (1 + (Po / 100)) * ALa + S

Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(line2D10)

Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(line2D13)

Dim constraint18 As Constraint
Set constraint18 = constraints3.AddBiEltCst(catCstTypeDistance, reference35, reference36)

constraint18.Mode = catCstModeDrivingDimension

Dim length10 As Length
Set length10 = constraint18.Dimension

length10.Value = Ap / 2

Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromObject(line2D13)

Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(line2D11)

Dim constraint19 As Constraint
Set constraint19 = constraints3.AddBiEltCst(catCstTypeDistance, reference37, reference38)

constraint19.Mode = catCstModeDrivingDimension

Dim length11 As Length
Set length11 = constraint19.Dimension

length11.Value = Ap

sketch3.CloseEdition

part1.InWorkObject = sketch3
part1.Update

' POCKET RESTRICCIÓN PASILLO
' POCKET RESTRICCIÓN PASILLO

Dim pocket2 As Pocket
Set pocket2 = shapeFactory1.AddNewPocket(sketch3, Pe)

part1.Update

' FIT ALL IN
' FIT ALL IN

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

' GUARDADO DEL PART
' GUARDADO DEL PART

```

```
Dim RutaPart As String
Dim MensajePrecaución As String
RutaPart = CATIA.FileSelectionBox("SaveAs", "*.CATPart", 1)
```

```
'Posible Cancelación
If RutaPart = "" Then
```

```
    MensajePrecaución = "La operación de guardado fue cancelada"
    MsgBox MensajePrecaución, 48, "Part no guardado"
    Exit Sub
```

```
End If
```

```
'Guardado
partDocument1.SaveAs RutaPart
```

```
.....
' CERRAMOS EL FORMULARIO CORRESPONDIENTE
.....
```

```
Carga1.Hide
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load EligePasillo1
Unload Me
EligePasillo1.Show
```

```
End Sub
```

- **Carga2.**

```
Private Sub CommandButton1_Click()
```

```
.....
' DEFINICIÓN DE VARIABLES
.....
```

```
Dim Oe1 As Double
Dim Po As Double
Dim Mr As Double
Dim Mr_ As Double
Dim H_s As Double
Dim S As Double
Dim Nf1 As Double
Dim ALa1 As Double
Dim Aa1 As Double
Dim La1 As Double
Dim P1 As Double
Dim SEa1 As Double
Dim Ap1 As Double
Dim Pe1 As Double
```

```
Oe1 = TextBox1.Value
Po = TextBox2.Value
Mr = TextBox3.Value
Mr_ = TextBox4.Value
H_s = TextBox5.Value
Nf1 = TextBox6.Value
NAfc1 = TextBox7.Value
ALa1 = TextBox8.Value
Aa1 = TextBox9.Value
La1 = TextBox10.Value
P1 = TextBox11.Value
```

```
SEa1 = TextBox12.Value
Ap1 = TextBox13.Value

S = Mr_ - H_s
Pe1 = Nf1 * La1 + (Nf1 - 1) * P1
Lt = NAfc1 * Aa1 + (NAfc1 - 1) * SEa1
```

```
.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....
```

```
Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body
```

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, Oe1#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

```
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
```

```
Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = -Oe1#
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 sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1
```

```

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

.....
' ESTANTES EN DOS PASILLOS
.....

Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(0#, 0#, Mr, Mr_, Mr, Mr_)

Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")

ellipse2D1.CenterPoint = point2D1

ellipse2D1.ReportName = 3

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference3)

constraint1.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint1.Dimension

length1.Value = Mr

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(ellipse2D1)

```



```

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference4)

constraint2.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint2.Dimension

length2.Value = Mr_

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(ellipse2D1)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeAngle, reference5, reference6)

constraint3.Mode = catCstModeDrivingDimension

constraint3.AngleSector = catCstAngleSector0

Dim angle1 As Angle
Set angle1 = constraint3.Dimension

angle1.Value = 0#

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

' PAD ESTANTES EN DOS PASILLOS
'
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, Pe1)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length3 As Length
Set length3 = limit1.Dimension

length3.Value = Pe1

part1.Update

' RESTRICCIÓN EN ALTURA
'
Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = -Oe1#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#

```

```

arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")

line2D3.ReportName = 1

Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")

line2D4.ReportName = 2

.....
' SKETCH RESTRICCIÓN EN ALTURA
.....

Dim point2D2 As Point2D
Set point2D2 = factory2D2.CreatePoint(-Mr, -Mr_)

point2D2.ReportName = 3

Dim point2D3 As Point2D
Set point2D3 = factory2D2.CreatePoint(-Mr, (1 + (Po / 100)) * ALa1 - S)

point2D3.ReportName = 4

Dim line2D5 As Line2D
Set line2D5 = factory2D2.CreateLine(-Mr, -Mr_, -Mr, (1 + (Po / 100)) * ALa1 - S)

```

```

line2D5.ReportName = 5

line2D5.StartPoint = point2D2

line2D5.EndPoint = point2D3

Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D5)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D4)

Dim constraint4 As Constraint
Set constraint4 = constraints2.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint4.Mode = catCstModeDrivingDimension

Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(Mr, (1 + (Po / 100)) * ALa1 - S)

point2D4.ReportName = 6

Dim line2D6 As Line2D
Set line2D6 = factory2D2.CreateLine(-Mr, (1 + (Po / 100)) * ALa1 - S, Mr, (1 + (Po / 100)) * ALa1 - S)

line2D6.ReportName = 7

line2D6.StartPoint = point2D3

line2D6.EndPoint = point2D4

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeHorizontal, reference9, reference10)

constraint5.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D
Set point2D5 = factory2D2.CreatePoint(Mr, -Mr_)

point2D5.ReportName = 8

Dim line2D7 As Line2D
Set line2D7 = factory2D2.CreateLine(Mr, (1 + (Po / 100)) * ALa1 - S, Mr, -Mr_)

line2D7.ReportName = 9

line2D7.StartPoint = point2D4

line2D7.EndPoint = point2D5

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D7)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D4)

Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeVerticality, reference11, reference12)

```

```

constraint6.Mode = catCstModeDrivingDimension

Dim line2D8 As Line2D
Set line2D8 = factory2D2.CreateLine(Mr, -Mr_, -Mr, -Mr_)

line2D8.ReportName = 10

line2D8.StartPoint = point2D5

line2D8.EndPoint = point2D2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D8)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)

Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference13, reference14)

constraint7.Mode = catCstModeDrivingDimension

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D3)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints2.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = (1 + (Po / 100)) * ALa1 - S

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D8)

Dim constraint9 As Constraint
Set constraint9 = constraints2.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint9.Mode = catCstModeDrivingDimension

Dim length5 As Length
Set length5 = constraint9.Dimension

length5.Value = Mr_ + (1 + (Po / 100)) * ALa1 - S

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D4)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D7)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeDistance, reference19, reference20)

constraint10.Mode = catCstModeDrivingDimension

Dim length6 As Length

```

```

Set length6 = constraint10.Dimension

length6.Value = Mr

Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D7)

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D5)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeDistance, reference21, reference22)

constraint11.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint11.Dimension

length7.Value = 2 * Mr

sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

' POCKET RESTRICCIÓN EN ALTURA
' POCKET RESTRICCIÓN EN ALTURA

Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch2, Pe1)

part1.Update

' RESTRICCIÓN PASILLOS
' RESTRICCIÓN PASILLOS

Dim sketch3 As Sketch
Set sketch3 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble3(8)
arrayOfVariantOfDouble3(0) = -0e1#
arrayOfVariantOfDouble3(1) = 0#
arrayOfVariantOfDouble3(2) = 0#
arrayOfVariantOfDouble3(3) = 0#
arrayOfVariantOfDouble3(4) = 1#
arrayOfVariantOfDouble3(5) = 0#
arrayOfVariantOfDouble3(6) = 0#
arrayOfVariantOfDouble3(7) = 0#
arrayOfVariantOfDouble3(8) = 1#
Set sketch3Variant = sketch3

' SISTEMA DE EJES ABSOLUTOS
' SISTEMA DE EJES ABSOLUTOS

sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3
part1.InWorkObject = sketch3

' CONJUNTO DE HERRAMIENTAS 2D
' CONJUNTO DE HERRAMIENTAS 2D

Dim factory2D3 As Factory2D
Set factory2D3 = sketch3.OpenEdition()

```

```
.....  
' ELEMENTOS GEOMÉTRICOS  
.....
```

```
Dim geometricElements3 As GeometricElements  
Set geometricElements3 = sketch3.GeometricElements
```

```
.....  
' SISTEMA DE EJES DENTRO DEL SKETCH  
.....
```

```
Dim axis2D3 As Axis2D  
Set axis2D3 = geometricElements3.Item("AbsoluteAxis")
```

```
.....  
' DIRECCIONES HORIZONTAL Y VERTICAL  
.....
```

```
Dim line2D9 As Line2D  
Set line2D9 = axis2D3.GetItem("HDirection")
```

```
line2D9.ReportName = 1
```

```
Dim line2D10 As Line2D  
Set line2D10 = axis2D3.GetItem("VDirection")
```

```
line2D10.ReportName = 2
```

```
.....  
' SKETCH RESTRICCIÓN PASILLOS  
.....
```

```
Dim point2D6 As Point2D  
Set point2D6 = factory2D3.CreatePoint(-((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S)
```

```
point2D6.ReportName = 3
```

```
Dim point2D7 As Point2D  
Set point2D7 = factory2D3.CreatePoint(-((Lt / 2) + Ap1), Mr_)
```

```
point2D7.ReportName = 4
```

```
Dim line2D11 As Line2D  
Set line2D11 = factory2D3.CreateLine(-((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S, -((Lt / 2) + Ap1), Mr_)
```

```
line2D11.ReportName = 5
```

```
line2D11.StartPoint = point2D6
```

```
line2D11.EndPoint = point2D7
```

```
Dim constraints3 As Constraints  
Set constraints3 = sketch3.Constraints
```

```
Dim reference23 As Reference  
Set reference23 = part1.CreateReferenceFromObject(line2D11)
```

```
Dim reference24 As Reference  
Set reference24 = part1.CreateReferenceFromObject(line2D10)
```

```
Dim constraint12 As Constraint  
Set constraint12 = constraints3.AddBiEltCst(catCstTypeVerticality, reference23, reference24)
```

```
constraint12.Mode = catCstModeDrivingDimension
```

```
Dim point2D8 As Point2D  
Set point2D8 = factory2D3.CreatePoint(-Lt / 2, Mr_)
```

```

point2D8.ReportName = 6

Dim line2D12 As Line2D
Set line2D12 = factory2D3.CreateLine(-((Lt / 2) + Ap1), Mr_, -Lt / 2, Mr_)

line2D12.ReportName = 7

line2D12.StartPoint = point2D7

line2D12.EndPoint = point2D8

Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(line2D12)

Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D9)

Dim constraint13 As Constraint
Set constraint13 = constraints3.AddBiEltCst(catCstTypeHorizontality, reference25, reference26)

constraint13.Mode = catCstModeDrivingDimension

Dim point2D9 As Point2D
Set point2D9 = factory2D3.CreatePoint(-Lt / 2, (1 + (Po / 100)) * ALa1 - S)

point2D9.ReportName = 8

Dim line2D13 As Line2D
Set line2D13 = factory2D3.CreateLine(-Lt / 2, Mr_, -Lt / 2, (1 + (Po / 100)) * ALa1 - S)

line2D13.ReportName = 9

line2D13.StartPoint = point2D8

line2D13.EndPoint = point2D9

Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D13)

Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D10)

Dim constraint14 As Constraint
Set constraint14 = constraints3.AddBiEltCst(catCstTypeVerticality, reference27, reference28)

constraint14.Mode = catCstModeDrivingDimension

Dim line2D14 As Line2D
Set line2D14 = factory2D3.CreateLine(-Lt / 2, (1 + (Po / 100)) * ALa1 - S, -((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S)

line2D14.ReportName = 10

line2D14.StartPoint = point2D9

line2D14.EndPoint = point2D6

Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(line2D14)

Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(line2D9)

Dim constraint15 As Constraint
Set constraint15 = constraints3.AddBiEltCst(catCstTypeHorizontality, reference29, reference30)

```

constraint15.Mode = catCstModeDrivingDimension

Dim reference31 As Reference
Set reference31 = part1.CreateReferenceFromObject(line2D9)

Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D14)

Dim constraint16 As Constraint
Set constraint16 = constraints3.AddBiEltCst(catCstTypeDistance, reference31, reference32)

constraint16.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint16.Dimension

length8.Value = (1 + (Po / 100)) * ALa1 - S

Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D14)

Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromObject(line2D12)

Dim constraint17 As Constraint
Set constraint17 = constraints3.AddBiEltCst(catCstTypeDistance, reference33, reference34)

constraint17.Mode = catCstModeDrivingDimension

Dim length9 As Length
Set length9 = constraint17.Dimension

length9.Value = Mr_ - (1 + (Po / 100)) * ALa1 + S

Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(line2D10)

Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(line2D13)

Dim constraint18 As Constraint
Set constraint18 = constraints3.AddBiEltCst(catCstTypeDistance, reference35, reference36)

constraint18.Mode = catCstModeDrivingDimension

Dim length10 As Length
Set length10 = constraint18.Dimension

length10.Value = Lt / 2

Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromObject(line2D13)

Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(line2D11)

Dim constraint19 As Constraint
Set constraint19 = constraints3.AddBiEltCst(catCstTypeDistance, reference37, reference38)

constraint19.Mode = catCstModeDrivingDimension

Dim length11 As Length
Set length11 = constraint19.Dimension

length11.Value = Ap1

sketch3.CloseEdition


```

part1.InWorkObject = sketch3
part1.Update

.....

' POCKET RESTRICCIÓN PASILLOS
.....

Dim pocket2 As Pocket
Set pocket2 = shapeFactory1.AddNewPocket(sketch3, Pe1)

part1.Update

.....

' RESTRICCIÓN PASILLOS 2
.....

Dim sketch4 As Sketch
Set sketch4 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble4(8)
arrayOfVariantOfDouble4(0) = -0e1#
arrayOfVariantOfDouble4(1) = 0#
arrayOfVariantOfDouble4(2) = 0#
arrayOfVariantOfDouble4(3) = 0#
arrayOfVariantOfDouble4(4) = 1#
arrayOfVariantOfDouble4(5) = 0#
arrayOfVariantOfDouble4(6) = 0#
arrayOfVariantOfDouble4(7) = 0#
arrayOfVariantOfDouble4(8) = 1#
Set sketch4Variant = sketch4

.....

' SISTEMA DE EJES ABSOLUTOS
.....

sketch4Variant.SetAbsoluteAxisData arrayOfVariantOfDouble4
part1.InWorkObject = sketch4

.....

' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D4 As Factory2D
Set factory2D4 = sketch4.OpenEdition()

.....

' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements4 As GeometricElements
Set geometricElements4 = sketch4.GeometricElements

.....

' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D4 As Axis2D
Set axis2D4 = geometricElements4.Item("AbsoluteAxis")

.....

' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D15 As Line2D
Set line2D15 = axis2D4.GetItem("HDirection")

```

```

line2D15.ReportName = 1

Dim line2D16 As Line2D
Set line2D16 = axis2D4.GetItem("VDirection")

line2D16.ReportName = 2

' SKETCH RESTRICCIÓN PASILLOS 2

Dim point2D10 As Point2D
Set point2D10 = factory2D4.CreatePoint(((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S)

point2D10.ReportName = 3

Dim point2D11 As Point2D
Set point2D11 = factory2D4.CreatePoint(((Lt / 2) + Ap1), Mr_)

point2D11.ReportName = 4

Dim line2D17 As Line2D
Set line2D17 = factory2D4.CreateLine(((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S, ((Lt / 2) + Ap1), Mr_)

line2D17.ReportName = 5

line2D17.StartPoint = point2D10

line2D17.EndPoint = point2D11

Dim constraints4 As Constraints
Set constraints4 = sketch4.Constraints

Dim reference39 As Reference
Set reference39 = part1.CreateReferenceFromObject(line2D17)

Dim reference40 As Reference
Set reference40 = part1.CreateReferenceFromObject(line2D16)

Dim constraint21 As Constraint
Set constraint21 = constraints4.AddBiEltCst(catCstTypeVerticality, reference39, reference40)

constraint21.Mode = catCstModeDrivingDimension

Dim point2D12 As Point2D
Set point2D12 = factory2D4.CreatePoint(Lt / 2, Mr_)

point2D12.ReportName = 6

Dim line2D18 As Line2D
Set line2D18 = factory2D4.CreateLine(((Lt / 2) + Ap1), Mr_, Lt / 2, Mr_)

line2D18.ReportName = 7

line2D18.StartPoint = point2D11

line2D18.EndPoint = point2D12

Dim reference41 As Reference
Set reference41 = part1.CreateReferenceFromObject(line2D18)

Dim reference42 As Reference
Set reference42 = part1.CreateReferenceFromObject(line2D15)

Dim constraint22 As Constraint
Set constraint22 = constraints4.AddBiEltCst(catCstTypeHorizontality, reference41, reference42)

```

```

constraint22.Mode = catCstModeDrivingDimension

Dim point2D13 As Point2D
Set point2D13 = factory2D4.CreatePoint(Lt / 2, (1 + (Po / 100)) * ALa1 - S)

point2D13.ReportName = 8

Dim line2D19 As Line2D
Set line2D19 = factory2D4.CreateLine(Lt / 2, Mr_, Lt / 2, (1 + (Po / 100)) * ALa1 - S)

line2D19.ReportName = 9

line2D19.StartPoint = point2D12

line2D19.EndPoint = point2D13

Dim reference43 As Reference
Set reference43 = part1.CreateReferenceFromObject(line2D19)

Dim reference44 As Reference
Set reference44 = part1.CreateReferenceFromObject(line2D16)

Dim constraint23 As Constraint
Set constraint23 = constraints4.AddBiEltCst(catCstTypeVerticality, reference43, reference44)

constraint23.Mode = catCstModeDrivingDimension

Dim line2D20 As Line2D
Set line2D20 = factory2D4.CreateLine(Lt / 2, (1 + (Po / 100)) * ALa1 - S, ((Lt / 2) + Ap1), (1 + (Po / 100)) * ALa1 - S)

line2D20.ReportName = 10

line2D20.StartPoint = point2D13

line2D20.EndPoint = point2D10

Dim reference45 As Reference
Set reference45 = part1.CreateReferenceFromObject(line2D20)

Dim reference46 As Reference
Set reference46 = part1.CreateReferenceFromObject(line2D15)

Dim constraint24 As Constraint
Set constraint24 = constraints4.AddBiEltCst(catCstTypeHorizontality, reference45, reference46)

constraint24.Mode = catCstModeDrivingDimension

Dim reference47 As Reference
Set reference47 = part1.CreateReferenceFromObject(line2D15)

Dim reference48 As Reference
Set reference48 = part1.CreateReferenceFromObject(line2D20)

Dim constraint25 As Constraint
Set constraint25 = constraints4.AddBiEltCst(catCstTypeDistance, reference47, reference48)

constraint25.Mode = catCstModeDrivingDimension

Dim length12 As Length
Set length12 = constraint25.Dimension

length12.Value = (1 + (Po / 100)) * ALa1 - S

Dim reference49 As Reference
Set reference49 = part1.CreateReferenceFromObject(line2D20)

```

```

Dim reference50 As Reference
Set reference50 = part1.CreateReferenceFromObject(line2D18)

Dim constraint26 As Constraint
Set constraint26 = constraints4.AddBiEltCst(catCstTypeDistance, reference49, reference50)

constraint26.Mode = catCstModeDrivingDimension

Dim length13 As Length
Set length13 = constraint26.Dimension

length13.Value = Mr_ - (1 + (Po / 100)) * ALa1 + S

Dim reference51 As Reference
Set reference51 = part1.CreateReferenceFromObject(line2D16)

Dim reference52 As Reference
Set reference52 = part1.CreateReferenceFromObject(line2D19)

Dim constraint27 As Constraint
Set constraint27 = constraints4.AddBiEltCst(catCstTypeDistance, reference51, reference52)

constraint27.Mode = catCstModeDrivingDimension

Dim length14 As Length
Set length14 = constraint27.Dimension

length14.Value = Lt / 2

Dim reference53 As Reference
Set reference53 = part1.CreateReferenceFromObject(line2D19)

Dim reference54 As Reference
Set reference54 = part1.CreateReferenceFromObject(line2D17)

Dim constraint28 As Constraint
Set constraint28 = constraints4.AddBiEltCst(catCstTypeDistance, reference53, reference54)

constraint28.Mode = catCstModeDrivingDimension

Dim length15 As Length
Set length15 = constraint28.Dimension

length15.Value = Ap1

sketch4.CloseEdition

part1.InWorkObject = sketch4
part1.Update

' POCKET RESTRICCIÓN PASILLOS
' POCKET RESTRICCIÓN PASILLOS

Dim pocket3 As Pocket
Set pocket3 = shapeFactory1.AddNewPocket(sketch4, Pe1)

part1.Update

' FIT ALL IN
' FIT ALL IN

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D

```

```

Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

' GUARDADO DEL PART
' GUARDADO DEL PART

Dim RutaPart As String
Dim MensajePrecaución As String
RutaPart = CATIA.FileSelectionBox("SaveAs", "*.CATPart", 1)

'Posible Cancelación
If RutaPart = "" Then

    MensajePrecaución = "La operación de guardado fue cancelada"
    MsgBox MensajePrecaución, 48, "Part no guardado"
    Exit Sub

End If

'Guardado
partDocument1.SaveAs RutaPart

' CERRAMOS EL FORMULARIO CORRESPONDIENTE

Carga2.Hide

End Sub

Private Sub CommandButton2_Click()

Load EligePasillo1
Unload Me
EligePasillo1.Show

End Sub

• Mercancía.

Private Sub CommandButton1_Click()

' DEFINICIÓN DE VARIABLES

Dim Om As Double
Dim A11 As Double
Dim An1 As Double
Dim L1 As Double
Dim Mr_ As Double
Dim H_s As Double
Dim S As Double

Om = TextBox1.Value
A11 = TextBox2.Value
An1 = TextBox3.Value
L1 = TextBox4.Value
Mr_ = TextBox5.Value
H_s = TextBox6.Value

S = Mr_ - H_s

```

.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....

Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, S#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")

body1.InsertHybridShape hybridShapePlaneOffset1
part1.InWorkObject = hybridShapePlaneOffset1

part1.Update

Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = -S#
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#

Set sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

```
.....  
' ELEMENTOS GEOMÉTRICOS  
.....
```

```
Dim geometricElements1 As GeometricElements  
Set geometricElements1 = sketch1.GeometricElements
```

```
.....  
' SISTEMA DE EJES DENTRO DEL SKETCH  
.....
```

```
Dim axis2D1 As Axis2D  
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
```

```
.....  
' DIRECCIONES HORIZONTAL Y VERTICAL  
.....
```

```
Dim line2D1 As Line2D  
Set line2D1 = axis2D1.GetItem("HDirection")
```

```
line2D1.ReportName = 1
```

```
Dim line2D2 As Line2D  
Set line2D2 = axis2D1.GetItem("VDirection")
```

```
line2D2.ReportName = 2
```

```
.....  
' MERCANCÍA  
.....
```

```
Dim point2D1 As Point2D  
Set point2D1 = factory2D1.CreatePoint(-(Om + L1), An1 / 2)
```

```
point2D1.ReportName = 3
```

```
Dim point2D2 As Point2D  
Set point2D2 = factory2D1.CreatePoint(-Om, An1 / 2)
```

```
point2D2.ReportName = 4
```

```
Dim line2D3 As Line2D  
Set line2D3 = factory2D1.CreateLine(-(Om + L1), An1 / 2, -Om, An1 / 2)
```

```
line2D3.ReportName = 5
```

```
line2D3.StartPoint = point2D1
```

```
line2D3.EndPoint = point2D2
```

```
Dim point2D3 As Point2D  
Set point2D3 = factory2D1.CreatePoint(-Om, -An1 / 2)
```

```
point2D3.ReportName = 6
```

```
Dim line2D4 As Line2D  
Set line2D4 = factory2D1.CreateLine(-Om, An1 / 2, -Om, -An1 / 2)
```

```
line2D4.ReportName = 7
```

```
line2D4.EndPoint = point2D2
```

```
line2D4.StartPoint = point2D3
```

```
Dim point2D4 As Point2D  
Set point2D4 = factory2D1.CreatePoint(-(Om + L1), -An1 / 2)
```

```

point2D4.ReportName = 8

Dim line2D5 As Line2D
Set line2D5 = factory2D1.CreateLine(-Om, -An1 / 2, -(Om + L1), -An1 / 2)

line2D5.ReportName = 9

line2D5.StartPoint = point2D3

line2D5.EndPoint = point2D4

Dim line2D6 As Line2D
Set line2D6 = factory2D1.CreateLine(-(Om + L1), -An1 / 2, -(Om + L1), An1 / 2)

line2D6.ReportName = 10

line2D6.EndPoint = point2D4

line2D6.StartPoint = point2D1

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D3)

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference3, reference4)

constraint1.Mode = catCstModeDrivingDimension

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D5)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference5, reference6)

constraint2.Mode = catCstModeDrivingDimension

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D4)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint3.Mode = catCstModeDrivingDimension

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D2)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeVerticality, reference9, reference10)

constraint4.Mode = catCstModeDrivingDimension

```



```

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D1)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeDistance, reference11, reference12)

constraint5.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint5.Dimension

length1.Value = An1 / 2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D3)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D5)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeDistance, reference13, reference14)

constraint6.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint6.Dimension

length2.Value = An1

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D2)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D4)

Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint7.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint7.Dimension

length3.Value = Om

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D4)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints1.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = L1

sketch1.CloseEdition

```

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

```
.....
' PAD MERCANCÍA
.....
```

```
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory
```

```
Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, All)
```

```
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
```

```
Dim length5 As Length
Set length5 = limit1.Dimension
```

```
length5.Value = All
```

```
part1.Update
```

```
.....
' FIT ALL IN
.....
```

```
Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow
```

```
Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer
```

```
viewer3D1.Reframe
```

```
Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D
```

```
.....
' GUARDADO DEL PART
.....
```

```
Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath
Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)
Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing
```

```
partDocument1.SaveAs objPath & "\\Mercancia.CATPart"
```

```
.....
' CERRAMOS EL FORMULARIO CORRESPONDIENTE
.....
```

```
Mercancia.Hide
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load Carga  
Unload Me  
Carga.Show
```

```
End Sub
```

- **Bodega.**

```
Private Sub CommandButton1_Click()
```

```
.....
```

```
' DEFINICIÓN DE VARIABLES
```

```
.....
```

```
Dim A_ As Double  
Dim A As Double  
Dim B_ As Double  
Dim B As Double  
Dim Pc As Double  
Dim Mr_ As Double  
Dim E_s As Double  
Dim H_s As Double  
Dim S As Double
```

```
A_ = TextBox1.Value  
A = TextBox2.Value  
B_ = TextBox3.Value  
B = TextBox4.Value  
Pc = TextBox5.Value  
Mr_ = TextBox6.Value  
H_s = TextBox7.Value  
E_s = TextBox8.Value
```

```
S = Mr_ - H_s
```

```
.....
```

```
' DEFINICIÓN DEL ENTORNO DE TRABAJO
```

```
.....
```

```
Dim documents1 As Documents  
Dim partDocument1 As PartDocument  
Dim part1 As Part  
Dim bodies1 As Bodies  
Dim body1 As Body  
Dim sketches1 As Sketches  
Dim originElements1 As OriginElements  
Dim reference1 As Reference  
Dim sketch1 As Sketch  
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 documents1 = CATIA.Documents  
Set partDocument1 = documents1.Add("Part")  
Set part1 = partDocument1.Part  
Set bodies1 = part1.Bodies
```

```

Set body1 = bodies1.Item("PartBody")
Set sketches1 = body1.Sketches
Set originElements1 = part1.OriginElements
Set reference1 = originElements1.PlaneYZ
Set sketch1 = sketches1.Add(reference1)
Set sketch1Variant = sketch1

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

.....
' CARGA EN LA BODEGA
.....

Dim point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(B_ / 2, -(S + E_s + A))

point2D1.ReportName = 3

Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(B / 2, -(S + E_s + A - A_))

point2D2.ReportName = 4

Dim line2D3 As Line2D
Set line2D3 = factory2D1.CreateLine(B_ / 2, -(S + E_s + A), B / 2, -(S + E_s + A - A_))

line2D3.ReportName = 5

```

```

line2D3.StartPoint = point2D1

line2D3.EndPoint = point2D2

Dim point2D3 As Point2D
Set point2D3 = factory2D1.CreatePoint(B / 2, -(S + E_s))

point2D3.ReportName = 6

Dim line2D4 As Line2D
Set line2D4 = factory2D1.CreateLine(B / 2, -(S + E_s + A - A_), B / 2, -(S + E_s))

line2D4.ReportName = 7

line2D4.StartPoint = point2D2

line2D4.EndPoint = point2D3

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference2 As Reference
Set reference2 = part1.CreateReferenceFromObject(line2D4)

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D2)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeVerticality, reference2, reference3)

constraint1.Mode = catCstModeDrivingDimension

Dim point2D4 As Point2D
Set point2D4 = factory2D1.CreatePoint(-B / 2, -(S + E_s))

point2D4.ReportName = 8

Dim line2D5 As Line2D
Set line2D5 = factory2D1.CreateLine(B / 2, -(S + E_s), -B / 2, -(S + E_s))

line2D5.ReportName = 9

line2D5.StartPoint = point2D3

line2D5.EndPoint = point2D4

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D5)

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference4, reference5)

constraint2.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D
Set point2D5 = factory2D1.CreatePoint(-B / 2, -(S + E_s + A - A_))

point2D5.ReportName = 10

Dim line2D6 As Line2D
Set line2D6 = factory2D1.CreateLine(-B / 2, -(S + E_s), -B / 2, -(S + E_s + A - A_))

line2D6.ReportName = 11

```

```

line2D6.StartPoint = point2D4

line2D6.EndPoint = point2D5

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D6)

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D2)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference6, reference7)

constraint3.Mode = catCstModeDrivingDimension

Dim point2D6 As Point2D
Set point2D6 = factory2D1.CreatePoint(-B_ / 2, -(S + E_s + A))

point2D6.ReportName = 12

Dim line2D7 As Line2D
Set line2D7 = factory2D1.CreateLine(-B / 2, -(S + E_s + A - A_), -B_ / 2, -(S + E_s + A))

line2D7.ReportName = 13

line2D7.StartPoint = point2D5

line2D7.EndPoint = point2D6

Dim line2D8 As Line2D
Set line2D8 = factory2D1.CreateLine(-B_ / 2, -(S + E_s + A), B_ / 2, -(S + E_s + A))

line2D8.ReportName = 14

line2D8.StartPoint = point2D6

line2D8.EndPoint = point2D1

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D8)

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D1)

Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference8, reference9)

constraint4.Mode = catCstModeDrivingDimension

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D1)

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(point2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeDistance, reference10, reference11)

constraint5.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint5.Dimension

length1.Value = S + E_s

Dim reference12 As Reference

```

```

Set reference12 = part1.CreateReferenceFromObject(point2D3)

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D8)

Dim constraint6 As Constraint
Set constraint6 = constraints1.AddBiEltCst(catCstTypeDistance, reference12, reference13)

constraint6.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint6.Dimension

length2.Value = A

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D2)

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(point2D3)

Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeDistance, reference14, reference15)

constraint7.Mode = catCstModeDrivingDimension

Dim length3 As Length
Set length3 = constraint7.Dimension

length3.Value = B / 2

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D4)

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints1.AddBiEltCst(catCstTypeDistance, reference16, reference17)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = B

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D2)

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(point2D1)

Dim constraint9 As Constraint
Set constraint9 = constraints1.AddBiEltCst(catCstTypeDistance, reference18, reference19)

constraint9.Mode = catCstModeDrivingDimension

Dim length5 As Length
Set length5 = constraint9.Dimension

length5.Value = B_ / 2

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D2)

Dim reference21 As Reference

```

```

Set reference21 = part1.CreateReferenceFromObject(point2D6)

Dim constraint10 As Constraint
Set constraint10 = constraints1.AddBiEltCst(catCstTypeDistance, reference20, reference21)

constraint10.Mode = catCstModeDrivingDimension

Dim length6 As Length
Set length6 = constraint10.Dimension

length6.Value = B_ / 2

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D4)

Dim constraint11 As Constraint
Set constraint11 = constraints1.AddMonoEltCst(catCstTypeLength, reference22)

constraint11.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint11.Dimension

length7.Value = A - A_

Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D6)

Dim constraint12 As Constraint
Set constraint12 = constraints1.AddMonoEltCst(catCstTypeLength, reference23)

constraint12.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint12.Dimension

length8.Value = A - A_

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

' PAD CARGA EN LA BODEGA

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, Pc)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length9 As Length
Set length9 = limit1.Dimension

length9.Value = Pc

part1.UpdateObject pad1
part1.Update

```



```
.....  
' DEFINICIÓN DEL ENTORNO DE TRABAJO  
.....
```

```
Dim specsAndGeomWindow1 As SpecsAndGeomWindow  
Set specsAndGeomWindow1 = CATIA.ActiveWindow
```

```
Dim viewer3D1 As Viewer3D  
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer
```

```
viewer3D1.Reframe
```

```
Dim viewpoint3D1 As Viewpoint3D  
Set viewpoint3D1 = viewer3D1.Viewpoint3D
```

```
.....  
' GUARDADO DEL PART  
.....
```

```
Const WINDOW_HANDLE = 0  
Const NO_OPTIONS = &H1  
Dim objShellApp  
Dim objFolder  
Dim objFldrItem  
Dim objPath  
Set objShellApp = CreateObject("Shell.Application")  
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)  
Set objFldrItem = objFolder.Self  
objPath = objFldrItem.Path  
BrowseForFolderDialogBox = objPath  
Set objShellApp = Nothing  
Set objFolder = Nothing  
Set objFldrItem = Nothing
```

```
partDocument1.SaveAs objPath & "\\Bodega.CATPart"
```

```
.....  
' CERRAMOS EL FORMULARIO CORRESPONDIENTE  
.....
```

```
Bodega.Hide  
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load VentanaPrincipal  
Unload Me  
VentanaPrincipal.Show
```

```
End Sub
```

- **Baño.**

```
Private Sub CommandButton1_Click()
```

```
.....  
' DEFINICIÓN DE VARIABLES  
.....
```

```
Dim Ob As Double  
Dim Ab As Double  
Dim Pb As Double  
Dim Mr_ As Double  
Dim H_s As Double  
Dim S As Double
```

```
Ob = TextBox1.Value
Ab = TextBox2.Value
Pb = TextBox3.Value
Mr = TextBox4.Value
Mr_ = TextBox5.Value
H_s = TextBox6.Value
```

```
S = Mr_ - H_s
```

```
.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....
```

```
Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body
```

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, Ob#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

```
body1.InsertHybridShape hybridShapePlaneOffset1
```

```
part1.InWorkObject = hybridShapePlaneOffset1
part1.Update
```

```
Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = -Ob#
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 sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1.Variant = sketch1
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

' CONJUNTO DE HERRAMIENTAS 2D

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

' ELEMENTOS GEOMÉTRICOS

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

' SISTEMA DE EJES DENTRO DEL SKETCH

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

' DIRECCIONES HORIZONTAL Y VERTICAL

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

' BAÑO

Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(0#, 0#, Mr, Mr_, Mr, Mr_)

Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")

ellipse2D1.CenterPoint = point2D1

ellipse2D1.ReportName = 3

Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference3)

constraint1.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint1.Dimension

length1.Value = Mr

```

```

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference4)

constraint2.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint2.Dimension

length2.Value = Mr_

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(ellipse2D1)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeAngle, reference5, reference6)

constraint3.Mode = catCstModeDrivingDimension

constraint3.AngleSector = catCstAngleSector0

Dim angle1 As Angle
Set angle1 = constraint3.Dimension

angle1.Value = 0#

sketch1.CloseEdition

part1.InWorkObject = sketch1
part1.Update

' PAD BAÑO

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, Pb)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

Dim length3 As Length
Set length3 = limit1.Dimension

length3.Value = Pb

part1.Update

' RESTRICCIÓN EN ALTURA

Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble2(8)

```

```
arrayOfVariantOfDouble2(0) = -Ob#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2
```

```
.....
' SISTEMA DE EJES ABSOLUTOS
.....
```

```
sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2
```

```
.....
' CONJUNTO DE HERRAMIENTAS 2D
.....
```

```
Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()
```

```
.....
' ELEMENTOS GEOMÉTRICOS
.....
```

```
Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements
```

```
.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....
```

```
Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
```

```
.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....
```

```
Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")
```

```
line2D3.ReportName = 1
```

```
Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")
```

```
line2D4.ReportName = 2
```

```
.....
' SKETCH RESTRICCIÓN EN ALTURA
.....
```

```
Dim point2D2 As Point2D
Set point2D2 = factory2D2.CreatePoint(-Mr, -Mr_)
```

```
point2D2.ReportName = 3
```

```
Dim point2D3 As Point2D
Set point2D3 = factory2D2.CreatePoint(-Mr, -S)
```

```
point2D3.ReportName = 4
```

```

Dim line2D5 As Line2D
Set line2D5 = factory2D2.CreateLine(-Mr, -Mr_, -Mr, -S)

line2D5.ReportName = 5

line2D5.StartPoint = point2D2

line2D5.EndPoint = point2D3

Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D5)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D4)

Dim constraint4 As Constraint
Set constraint4 = constraints2.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint4.Mode = catCstModeDrivingDimension

Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(Mr, -S)

point2D4.ReportName = 6

Dim line2D6 As Line2D
Set line2D6 = factory2D2.CreateLine(-Mr, -S, Mr, -S)

line2D6.ReportName = 7

line2D6.StartPoint = point2D3

line2D6.EndPoint = point2D4

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference9, reference10)

constraint5.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D
Set point2D5 = factory2D2.CreatePoint(Mr, -Mr_)

point2D5.ReportName = 8

Dim line2D7 As Line2D
Set line2D7 = factory2D2.CreateLine(Mr, -S, Mr, -Mr_)

line2D7.ReportName = 9

line2D7.StartPoint = point2D4

line2D7.EndPoint = point2D5

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D7)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D4)

```

```

Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeVerticality, reference11, reference12)

constraint6.Mode = catCstModeDrivingDimension

Dim line2D8 As Line2D
Set line2D8 = factory2D2.CreateLine(Mr, -Mr_, -Mr, -Mr_)

line2D8.ReportName = 10

line2D8.StartPoint = point2D5

line2D8.EndPoint = point2D2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D8)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)

Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeHorizontal, reference13, reference14)

constraint7.Mode = catCstModeDrivingDimension

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D3)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints2.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = S

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D8)

Dim constraint9 As Constraint
Set constraint9 = constraints2.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint9.Mode = catCstModeDrivingDimension

Dim length5 As Length
Set length5 = constraint9.Dimension

length5.Value = Mr_ - S

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D4)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D7)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeDistance, reference19, reference20)

```

```

constraint10.Mode = catCstModeDrivingDimension

Dim length6 As Length
Set length6 = constraint10.Dimension

length6.Value = Mr

Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D7)

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D5)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeDistance, reference21, reference22)

constraint11.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint11.Dimension

length7.Value = 2 * Mr

sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

' POCKET RESTRICCIÓN EN ALTURA
' POCKET RESTRICCIÓN EN ALTURA

Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch2, Pb)

part1.Update

' RESTRICCIÓN EN ANCHURA
' RESTRICCIÓN EN ANCHURA

Dim sketch3 As Sketch
Set sketch3 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble3(8)
arrayOfVariantOfDouble3(0) = -Ob#
arrayOfVariantOfDouble3(1) = 0#
arrayOfVariantOfDouble3(2) = 0#
arrayOfVariantOfDouble3(3) = 0#
arrayOfVariantOfDouble3(4) = 1#
arrayOfVariantOfDouble3(5) = 0#
arrayOfVariantOfDouble3(6) = 0#
arrayOfVariantOfDouble3(7) = 0#
arrayOfVariantOfDouble3(8) = 1#
Set sketch3Variant = sketch3

' SISTEMA DE EJES ABSOLUTOS
' SISTEMA DE EJES ABSOLUTOS

sketch3Variant.SetAbsoluteAxisData arrayOfVariantOfDouble3
part1.InWorkObject = sketch3

' CONJUNTO DE HERRAMIENTAS 2D
' CONJUNTO DE HERRAMIENTAS 2D

```



```

Dim factory2D3 As Factory2D
Set factory2D3 = sketch3.OpenEdition()

' ELEMENTOS GEOMÉTRICOS

Dim geometricElements3 As GeometricElements
Set geometricElements3 = sketch3.GeometricElements

' SISTEMA DE EJES DENTRO DEL SKETCH

Dim axis2D3 As Axis2D
Set axis2D3 = geometricElements3.Item("AbsoluteAxis")

' DIRECCIONES HORIZONTAL Y VERTICAL

Dim line2D9 As Line2D
Set line2D9 = axis2D3.GetItem("HDirection")

line2D9.ReportName = 5

Dim line2D10 As Line2D
Set line2D10 = axis2D3.GetItem("VDirection")

line2D10.ReportName = 6

' SKETCH RESTRICCIÓN EN ANCHURA

Dim point2D6 As Point2D
Set point2D6 = factory2D3.CreatePoint(-(Mr - Ab), -S)

point2D6.ReportName = 7

Dim point2D7 As Point2D
Set point2D7 = factory2D3.CreatePoint(-(Mr - Ab), Mr_)

point2D7.ReportName = 8

Dim line2D11 As Line2D
Set line2D11 = factory2D3.CreateLine(-(Mr - Ab), -S, -(Mr - Ab), Mr_)

line2D11.ReportName = 1

line2D11.StartPoint = point2D6

line2D11.EndPoint = point2D7

Dim constraints3 As Constraints
Set constraints3 = sketch3.Constraints

Dim reference23 As Reference
Set reference23 = part1.CreateReferenceFromObject(line2D11)

Dim reference24 As Reference
Set reference24 = part1.CreateReferenceFromObject(line2D10)

Dim constraint12 As Constraint
Set constraint12 = constraints3.AddBiEltCst(catCstTypeVerticality, reference23, reference24)

constraint12.Mode = catCstModeDrivingDimension

```

```

Dim point2D8 As Point2D
Set point2D8 = factory2D3.CreatePoint(Mr, Mr_)

point2D8.ReportName = 9

Dim line2D12 As Line2D
Set line2D12 = factory2D3.CreateLine(-(Mr - Ab), Mr_, Mr, Mr_)

line2D12.ReportName = 2

line2D12.StartPoint = point2D7

line2D12.EndPoint = point2D8

Dim reference25 As Reference
Set reference25 = part1.CreateReferenceFromObject(line2D12)

Dim reference26 As Reference
Set reference26 = part1.CreateReferenceFromObject(line2D9)

Dim constraint13 As Constraint
Set constraint13 = constraints3.AddBiEltCst(catCstTypeHorizontality, reference25, reference26)

constraint13.Mode = catCstModeDrivingDimension

Dim point2D9 As Point2D
Set point2D9 = factory2D3.CreatePoint(Mr, -S)

point2D9.ReportName = 10

Dim line2D13 As Line2D
Set line2D13 = factory2D3.CreateLine(Mr, Mr_, Mr, -S)

line2D13.ReportName = 3

line2D13.StartPoint = point2D8

line2D13.EndPoint = point2D9

Dim reference27 As Reference
Set reference27 = part1.CreateReferenceFromObject(line2D13)

Dim reference28 As Reference
Set reference28 = part1.CreateReferenceFromObject(line2D10)

Dim constraint14 As Constraint
Set constraint14 = constraints3.AddBiEltCst(catCstTypeVerticality, reference27, reference28)

constraint14.Mode = catCstModeDrivingDimension

Dim line2D14 As Line2D
Set line2D14 = factory2D3.CreateLine(Mr, -S, -(Mr - Ab), -S)

line2D14.ReportName = 4

line2D14.StartPoint = point2D9

line2D14.EndPoint = point2D6

Dim reference29 As Reference
Set reference29 = part1.CreateReferenceFromObject(line2D14)

Dim reference30 As Reference
Set reference30 = part1.CreateReferenceFromObject(line2D9)

Dim constraint15 As Constraint

```

```

Set constraint15 = constraints3.AddBiEltCst(catCstTypeHorizontality, reference29, reference30)

constraint15.Mode = catCstModeDrivingDimension

Dim reference31 As Reference
Set reference31 = part1.CreateReferenceFromObject(line2D9)

Dim reference32 As Reference
Set reference32 = part1.CreateReferenceFromObject(line2D14)

Dim constraint16 As Constraint
Set constraint16 = constraints3.AddBiEltCst(catCstTypeDistance, reference31, reference32)

constraint16.Mode = catCstModeDrivingDimension

Dim length8 As Length
Set length8 = constraint16.Dimension

length8.Value = S

Dim reference33 As Reference
Set reference33 = part1.CreateReferenceFromObject(line2D14)

Dim reference34 As Reference
Set reference34 = part1.CreateReferenceFromObject(line2D12)

Dim constraint17 As Constraint
Set constraint17 = constraints3.AddBiEltCst(catCstTypeDistance, reference33, reference34)

constraint17.Mode = catCstModeDrivingDimension

Dim length9 As Length
Set length9 = constraint17.Dimension

length9.Value = Mr_ + S

Dim reference35 As Reference
Set reference35 = part1.CreateReferenceFromObject(line2D10)

Dim reference36 As Reference
Set reference36 = part1.CreateReferenceFromObject(line2D11)

Dim constraint18 As Constraint
Set constraint18 = constraints3.AddBiEltCst(catCstTypeDistance, reference35, reference36)

constraint18.Mode = catCstModeDrivingDimension

Dim length10 As Length
Set length10 = constraint18.Dimension

length10.Value = Mr - Ab

Dim reference37 As Reference
Set reference37 = part1.CreateReferenceFromObject(line2D11)

Dim reference38 As Reference
Set reference38 = part1.CreateReferenceFromObject(line2D13)

Dim constraint19 As Constraint
Set constraint19 = constraints3.AddBiEltCst(catCstTypeDistance, reference37, reference38)

constraint19.Mode = catCstModeDrivingDimension

Dim length11 As Length
Set length11 = constraint19.Dimension

length11.Value = 2 * Mr - Ab

```

```

sketch3.CloseEdition

part1.InWorkObject = sketch3
part1.Update

' POCKET RESTRICCIÓN EN ANCHURA
' POCKET RESTRICCIÓN EN ANCHURA

Dim pocket2 As Pocket
Set pocket2 = shapeFactory1.AddNewPocket(sketch3, Pb)

part1.Update

' FIT ALL IN
' FIT ALL IN

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

' GUARDADO DEL PART
' GUARDADO DEL PART

Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath
Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)
Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing

partDocument1.SaveAs objPath & "\Aseo.CATPart"

' CERRAMOS EL FORMULARIO CORRESPONDIENTE
' CERRAMOS EL FORMULARIO CORRESPONDIENTE

Baño.Hide
VentanaPrincipal.Show

End Sub

Private Sub CommandButton2_Click()

Load VentanaPrincipal
Unload Me
VentanaPrincipal.Show

End Sub

```

- **Galley.**

```
Private Sub CommandButton1_Click()
```

```
.....
' DEFINICIÓN DE VARIABLES
.....
```

```
Dim Og As Double
Dim P1 As Double
Dim L As Double
Dim Mr_ As Double
Dim H_s As Double
Dim S As Double
```

```
Og = TextBox1.Value
P1 = TextBox2.Value
Mr = TextBox3.Value
Mr_ = TextBox4.Value
H_s = TextBox5.Value
```

```
S = Mr_ - H_s
```

```
.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....
```

```
Dim documents1 As Documents
Dim partDocument1 As PartDocument
Dim part1 As Part
Dim hybridShapeFactory1 As HybridShapeFactory
Dim originElements1 As OriginElements
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Dim reference1 As Reference
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Dim bodies1 As Bodies
Dim body1 As Body
```

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Add("Part")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set originElements1 = part1.OriginElements
Set hybridShapePlaneExplicit1 = originElements1.PlaneYZ
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, Og#, True)
Set bodies1 = part1.Bodies
Set body1 = bodies1.Item("PartBody")
```

```
body1.InsertHybridShape hybridShapePlaneOffset1
part1.InWorkObject = hybridShapePlaneOffset1
```

```
part1.Update
```

```
Dim sketches1 As Sketches
Dim hybridShapes1 As HybridShapes
Dim reference2 As Reference
Dim sketch1 As Sketch
Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = -Og#
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 sketches1 = body1.Sketches
Set hybridShapes1 = body1.HybridShapes
Set reference2 = hybridShapes1.Item("Plane.1")
Set sketch1 = sketches1.Add(reference2)
Set sketch1Variant = sketch1

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1
part1.InWorkObject = sketch1

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D1 As Axis2D
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D1 As Line2D
Set line2D1 = axis2D1.GetItem("HDirection")

line2D1.ReportName = 1

Dim line2D2 As Line2D
Set line2D2 = axis2D1.GetItem("VDirection")

line2D2.ReportName = 2

.....
' GALLEY
.....

Dim ellipse2D1 As Ellipse2D
Set ellipse2D1 = factory2D1.CreateClosedEllipse(0#, 0#, Mr, Mr_, Mr, Mr_)

Dim point2D1 As Point2D
Set point2D1 = axis2D1.GetItem("Origin")

ellipse2D1.CenterPoint = point2D1

ellipse2D1.ReportName = 3

Dim constraints1 As Constraints

```

```

Set constraints1 = sketch1.Constraints

Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeMajorRadius, reference3)

constraint1.Mode = catCstModeDrivingDimension

Dim length1 As Length
Set length1 = constraint1.Dimension

length1.Value = Mr

Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(ellipse2D1)

Dim constraint2 As Constraint
Set constraint2 = constraints1.AddMonoEltCst(catCstTypeMinorRadius, reference4)

constraint2.Mode = catCstModeDrivingDimension

Dim length2 As Length
Set length2 = constraint2.Dimension

length2.Value = Mr_

Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(ellipse2D1)

Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)

Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeAngle, reference5, reference6)

constraint3.Mode = catCstModeDrivingDimension

constraint3.AngleSector = catCstAngleSector0

Dim angle1 As Angle
Set angle1 = constraint3.Dimension

angle1.Value = 0#

sketch1.CloseEdition

part1.InWorkObject = sketch1

part1.Update

.....
' PAD GALLEY
.....

Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory

Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, P1)

pad1.DirectionOrientation = catInverseOrientation

Dim limit1 As Limit
Set limit1 = pad1.FirstLimit

```

Dim length3 As Length
Set length3 = limit1.Dimension

length3.Value = P1

part1.Update

.....
' RESTRICCIÓN EN ALTURA
.....

Dim sketch2 As Sketch
Set sketch2 = sketches1.Add(reference2)

Dim arrayOfVariantOfDouble2(8)
arrayOfVariantOfDouble2(0) = -Og#
arrayOfVariantOfDouble2(1) = 0#
arrayOfVariantOfDouble2(2) = 0#
arrayOfVariantOfDouble2(3) = 0#
arrayOfVariantOfDouble2(4) = 1#
arrayOfVariantOfDouble2(5) = 0#
arrayOfVariantOfDouble2(6) = 0#
arrayOfVariantOfDouble2(7) = 0#
arrayOfVariantOfDouble2(8) = 1#
Set sketch2Variant = sketch2

.....
' SISTEMA DE EJES ABSOLUTOS
.....

sketch2Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
part1.InWorkObject = sketch2

.....
' CONJUNTO DE HERRAMIENTAS 2D
.....

Dim factory2D2 As Factory2D
Set factory2D2 = sketch2.OpenEdition()

.....
' ELEMENTOS GEOMÉTRICOS
.....

Dim geometricElements2 As GeometricElements
Set geometricElements2 = sketch2.GeometricElements

.....
' SISTEMA DE EJES DENTRO DEL SKETCH
.....

Dim axis2D2 As Axis2D
Set axis2D2 = geometricElements2.Item("AbsoluteAxis")

.....
' DIRECCIONES HORIZONTAL Y VERTICAL
.....

Dim line2D3 As Line2D
Set line2D3 = axis2D2.GetItem("HDirection")

line2D3.ReportName = 1

Dim line2D4 As Line2D
Set line2D4 = axis2D2.GetItem("VDirection")

line2D4.ReportName = 2

.....
' SKETCH RESTRICCIÓN EN ALTURA
.....

Dim point2D2 As Point2D
Set point2D2 = factory2D2.CreatePoint(-Mr, -Mr_)

point2D2.ReportName = 3

Dim point2D3 As Point2D
Set point2D3 = factory2D2.CreatePoint(-Mr, -S)

point2D3.ReportName = 4

Dim line2D5 As Line2D
Set line2D5 = factory2D2.CreateLine(-Mr, -Mr_, -Mr, -S)

line2D5.ReportName = 5

line2D5.StartPoint = point2D2

line2D5.EndPoint = point2D3

Dim constraints2 As Constraints
Set constraints2 = sketch2.Constraints

Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D5)

Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D4)

Dim constraint4 As Constraint
Set constraint4 = constraints2.AddBiEltCst(catCstTypeVerticality, reference7, reference8)

constraint4.Mode = catCstModeDrivingDimension

Dim point2D4 As Point2D
Set point2D4 = factory2D2.CreatePoint(Mr, -S)

point2D4.ReportName = 6

Dim line2D6 As Line2D
Set line2D6 = factory2D2.CreateLine(-Mr, -S, Mr, -S)

line2D6.ReportName = 7

line2D6.StartPoint = point2D3

line2D6.EndPoint = point2D4

Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)

Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D3)

Dim constraint5 As Constraint
Set constraint5 = constraints2.AddBiEltCst(catCstTypeHorizontalidad, reference9, reference10)

constraint5.Mode = catCstModeDrivingDimension

Dim point2D5 As Point2D

```

Set point2D5 = factory2D2.CreatePoint(Mr, -Mr_)

point2D5.ReportName = 8

Dim line2D7 As Line2D
Set line2D7 = factory2D2.CreateLine(Mr, -S, Mr, -Mr_)

line2D7.ReportName = 9

line2D7.StartPoint = point2D4

line2D7.EndPoint = point2D5

Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D7)

Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D4)

Dim constraint6 As Constraint
Set constraint6 = constraints2.AddBiEltCst(catCstTypeVerticality, reference11, reference12)

constraint6.Mode = catCstModeDrivingDimension

Dim line2D8 As Line2D
Set line2D8 = factory2D2.CreateLine(Mr, -Mr_, -Mr, -Mr_)

line2D8.ReportName = 10

line2D8.StartPoint = point2D5

line2D8.EndPoint = point2D2

Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D8)

Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)

Dim constraint7 As Constraint
Set constraint7 = constraints2.AddBiEltCst(catCstTypeHorizontality, reference13, reference14)

constraint7.Mode = catCstModeDrivingDimension

Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D3)

Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D6)

Dim constraint8 As Constraint
Set constraint8 = constraints2.AddBiEltCst(catCstTypeDistance, reference15, reference16)

constraint8.Mode = catCstModeDrivingDimension

Dim length4 As Length
Set length4 = constraint8.Dimension

length4.Value = S

Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)

Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D8)

Dim constraint9 As Constraint

```

```

Set constraint9 = constraints2.AddBiEltCst(catCstTypeDistance, reference17, reference18)

constraint9.Mode = catCstModeDrivingDimension

Dim length5 As Length
Set length5 = constraint9.Dimension

length5.Value = Mr_ - S

Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D4)

Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D7)

Dim constraint10 As Constraint
Set constraint10 = constraints2.AddBiEltCst(catCstTypeDistance, reference19, reference20)

constraint10.Mode = catCstModeDrivingDimension

Dim length6 As Length
Set length6 = constraint10.Dimension

length6.Value = Mr

Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D7)

Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D5)

Dim constraint11 As Constraint
Set constraint11 = constraints2.AddBiEltCst(catCstTypeDistance, reference21, reference22)

constraint11.Mode = catCstModeDrivingDimension

Dim length7 As Length
Set length7 = constraint11.Dimension

length7.Value = 2 * Mr

sketch2.CloseEdition

part1.InWorkObject = sketch2
part1.Update

' POCKET RESTRICCIÓN EN ALTURA
' POCKET RESTRICCIÓN EN ALTURA

Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch2, P1)

part1.Update

' FIT ALL IN
' FIT ALL IN

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

```

```
Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D
```

```
.....
' GUARDADO DEL PART
.....
```

```
Const WINDOW_HANDLE = 0
Const NO_OPTIONS = &H1
Dim objShellApp
Dim objFolder
Dim objFldrItem
Dim objPath
Set objShellApp = CreateObject("Shell.Application")
Set objFolder = objShellApp.BrowseForFolder(WINDOW_HANDLE, strTitle, NO_OPTIONS)
Set objFldrItem = objFolder.Self
objPath = objFldrItem.Path
BrowseForFolderDialogBox = objPath
Set objShellApp = Nothing
Set objFolder = Nothing
Set objFldrItem = Nothing
```

```
partDocument1.SaveAs objPath & "\Galley.CATPart"
```

```
.....
' CERRAMOS EL FORMULARIO CORRESPONDIENTE
.....
```

```
Galley.Hide
VentanaPrincipal.Show
```

```
End Sub
```

```
Private Sub CommandButton2_Click()
```

```
Load VentanaPrincipal
Unload Me
VentanaPrincipal.Show
```

```
End Sub
```

- **ConfirmaProducto.**

```
Private Sub CommandButton1_Click()
```

```
.....
' DEFINICIÓN DEL ENTORNO DE TRABAJO
.....
```

```
Dim documents1 As Documents
Dim productDocument1 As ProductDocument
Dim product1 As Product
Dim products1 As Products
```

```
Set documents1 = CATIA.Documents
Set productDocument1 = documents1.Add("Product")
Set product1 = productDocument1.Product
Set products1 = product1.Products
```

```
.....
' INSERCIÓN DE PARTS
.....
```

```
Dim Ruta As String
```

```
Ruta = TextBox1.Text
```

```

Dim arrayOfVariantOfBSTR1(0)
arrayOfVariantOfBSTR1(0) = Ruta & "\\SeccionCentral.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR1, "All"

Dim constraints1 As Constraints
Set constraints1 = product1.Connections("CATIAConstraints")

Dim reference1 As Reference
Set reference1 = product1.CreateReferenceFromName("Product1/Part1.1/!Product1/Part1.1/")

Dim constraint1 As Constraint
Set constraint1 = constraints1.AddMonoEltCst(catCstTypeReference, reference1)

If CheckBox2.Value = True Then

    Dim arrayOfVariantOfBSTR2(0)
    arrayOfVariantOfBSTR2(0) = Ruta & "\\Clase1.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR2, "All"

End If

If CheckBox3.Value = True Then

    Dim arrayOfVariantOfBSTR3(0)
    arrayOfVariantOfBSTR3(0) = Ruta & "\\Clase2.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR3, "All"

End If

If CheckBox4.Value = True Then

    Dim arrayOfVariantOfBSTR4(0)
    arrayOfVariantOfBSTR4(0) = Ruta & "\\Clase3.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR4, "All"

End If

If CheckBox5.Value = True Then

    Dim arrayOfVariantOfBSTR5(0)
    arrayOfVariantOfBSTR5(0) = Ruta & "\\Pasajero.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR5, "All"

End If

If CheckBox6.Value = True Then

    Dim arrayOfVariantOfBSTR6(0)
    arrayOfVariantOfBSTR6(0) = Ruta & "\\Carga1.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR6, "All"

End If

If CheckBox7.Value = True Then

    Dim arrayOfVariantOfBSTR7(0)
    arrayOfVariantOfBSTR7(0) = Ruta & "\\Carga2.CATPart"
    Set products1Variant = products1
    products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR7, "All"

```

End If

If CheckBox8.Value = True Then

```
Dim arrayOfVariantOfBSTR8(0)
arrayOfVariantOfBSTR8(0) = Ruta & "\Carga3.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR8, "All"
```

End If

If CheckBox9.Value = True Then

```
Dim arrayOfVariantOfBSTR9(0)
arrayOfVariantOfBSTR9(0) = Ruta & "\Carga4.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR9, "All"
```

End If

If CheckBox10.Value = True Then

```
Dim arrayOfVariantOfBSTR10(0)
arrayOfVariantOfBSTR10(0) = Ruta & "\Carga5.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR10, "All"
```

End If

If CheckBox11.Value = True Then

```
Dim arrayOfVariantOfBSTR11(0)
arrayOfVariantOfBSTR11(0) = Ruta & "\Mercancia.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR11, "All"
```

End If

If CheckBox12.Value = True Then

```
Dim arrayOfVariantOfBSTR12(0)
arrayOfVariantOfBSTR12(0) = Ruta & "\Carga.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR12, "All"
```

End If

If CheckBox13.Value = True Then

```
Dim arrayOfVariantOfBSTR13(0)
arrayOfVariantOfBSTR13(0) = Ruta & "\Aseo.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR13, "All"
```

End If

If CheckBox14.Value = True Then

```
Dim arrayOfVariantOfBSTR14(0)
arrayOfVariantOfBSTR14(0) = Ruta & "\Galley.CATPart"
Set products1Variant = products1
products1Variant.AddComponentsFromFiles arrayOfVariantOfBSTR14, "All"
```

End If

.....

' FIT ALL IN

.....

Dim specsAndGeomWindow1 As SpecsAndGeomWindow
Set specsAndGeomWindow1 = CATIA.ActiveWindow

Dim viewer3D1 As Viewer3D
Set viewer3D1 = specsAndGeomWindow1.ActiveViewer

viewer3D1.Reframe

Dim viewpoint3D1 As Viewpoint3D
Set viewpoint3D1 = viewer3D1.Viewpoint3D

.....

' ANÁLISIS DE INTERFERENCIAS

.....

Dim Clashes As Clashes
Set Clashes = CATIA.ActiveDocument.Product.GetTechnologicalObject("Clashes")

Dim Clash As Clash
Set Clash = Clashes.Add()

Clash.ComputationType = catClashComputationTypeBetweenAll
Clash.Compute
Clash.Export CatClashExportTypeXMLResultOnly, Ruta & "\Análisis.xml"

.....

' CERRAMOS EL FORMULARIO CORRESPONDIENTE

.....

ConformaProducto.Hide
VentanaPrincipal.Show

End Sub

Private Sub CommandButton2_Click()

Load VentanaPrincipal
Unload Me
VentanaPrincipal.Show

End Sub