Simulación de objetos

Aplicaciones por *Salome Meca, Code Saturne y ParaView*

COMUNICACIÓN CIENTÍFICA

Mario Ibañez Olvera
Alien Blanco Flores

Simulación de objetos
Aplicaciones por Salome Meca, Code Saturne y ParaView

Mario Ibañez Olvera
Alien Blanco Flores

COMUNICACIÓN
CIENTÍFICA

COMUNICACIÓN CIENTÍFICA PUBLICACIONES ARBITRADAS
HUMANIDADES, SOCIALES Y CIENCIAS

CDC
COLECCIÓN DIVULGACIÓN CIENTÍFICA

Cada libro de la Colección Divulgación Científica es evaluado para su publicación mediante el sistema de dictaminación de pares externos. Invitamos a ver el proceso de dictaminación transparentado, así como la consulta del libro en Acceso Abierto en

DOI.ORG/ 10.52501/cc.095

www.comunicacion-cientifica.com

Ediciones Comunicación Científica se especializa en la publicación de conocimiento científico en español e inglés en soporte de libro impreso y digital en las áreas de humanidades, ciencias sociales y ciencias exactas. Guía su criterio de publicación cumpliendo con las prácticas internacionales: dictaminación de pares ciegos externos, comités y ética editorial, verificación antiplagio, acceso abierto, medición del impacto de la publicación, difusión, distribución impresa y digital, transparencia editorial e indexación internacional.

Simulación de objetos

Aplicaciones por Salome Meca, Code Saturne y ParaView

Mario Ibañez Olvera
Alien Blanco Flores

COMUNICACIÓN
CIENTÍFICA

Ibañez Olvera, Mario

Simulación de objetos : aplicaciones por Salome Meca, Code Saturne y ParaView / Mario Ibañez Olvera, Alien Blanco Flores. Estado de México : Universidad Autónoma del Estado de México ; Cuidad de México : Comunicación Científica, 2023.

197 páginas : ilustraciones. – (Colección Divulgación Científica)

ISBN 978-607-633-562-8 (PDF Universidad Autónoma del Estado de México)

ISBN 978-607-59473-8-9 (PDF Ediciones Comunicación Científica)

DOI 10.52501/cc.095

1. Simulación por computadora — Estudio y enseñanza. I. Blanco Flores, Alien, autor. II. Título. III. Serie.

LC: QA76.9.C65 Dewey: 003.35369

Primera edición: Enero de 2023,
Simulación de objetos. Aplicaciones por Salome Meca, Code Saturne y ParaView
Mario Ibañez Olvera y Alien Blanco Flores

Libro sometido a sistema antiplagio y publicado con la previa revisión y aprobación de pares doble ciego externos, pertenecientes al Sistema Nacional de Investigadores, Nivel I. Expediente de obra 20/09/2015, Dirección de Difusión y Promoción de la Investigación y los Estudios Avanzados de la Universidad Autónoma del Estado de México.

ISBN 978-607-633-562-8 (PDF Universidad Autónoma del Estado de México)
ISBN 978-607-59473-8-9 (PDF Ediciones Comunicación Científica S.A. de C.V.)

Índice

Índice de figuras

Prólogo

Cuando se habla de software es necesario pensar en las diversas aplicaciones científicas, cuyo desarrollo se ha ido perfeccionando a través de los años, y su uso se ha diversificado, tanto en la industria como en las áreas de investigación. Hay en el mercado y en la red de internet, licencias de uso privado y licencias de uso libre.[1]

Los programas con licencia privada son producto de un progreso paulatino que ha logrado mejorar las aplicaciones en áreas de estudio variadas: matemáticas, química, electrónica, mecánica, mecánica molecular, medio ambiente, arquitectura, por citar algunas; esto pareciera ser benéfico para los usuarios por los avances paulatinos de la tecnología, pero tales mejoras representan un costo económico que aumenta día con día.

Ese motivo ha llevado a particulares y a organizaciones sociales a crear software libre en una panoplia de sectores e intereses. Una gama fundamental de los programas abiertos es la enfocada a crear modelos simulados de objetos, situaciones y escenarios, a través de la informática. Tales programas son cada vez más socorridos en la industria y en la investigación.

¿Qué son los modelos de simulación? Aquellos que "te permiten desarrollar un modelo informático de tu instalación real y, posteriormente, utilizar ese modelo para simular su funcionamiento a lo largo del tiempo. Esta

[1] Para fines del presente libro se considerarán los términos código abierto, programa abierto o formato libre para referir los programas de acceso sin costo.

información tiene un valor incalculable, ya que permite responder a importantes preguntas sobre el sistema sin necesidad de interrumpirlo". [5]

Seguramente en otras épocas se debieron gastar recursos exorbitantes para perfeccionar un producto o tal vez hubo que sacrificar a seres vivos para testear un cosmético o un fármaco. En la industria 4.0 el camino de la incertidumbre o la explotación de recursos naturales puede quedar para la historia. Por ello, el valor de la modelación es inestimable en el área del conocimiento o la empresa que recurra a sus beneficios. Este es el objeto del presente libro, ofrecer una aproximación a una pequeña rama de la modelación.

Los sistemas informáticos de modelado hacen las veces de un gemelo virtual para integrar la realidad que se espera en un futuro, antes de tomar decisiones en el mundo tangible. De ahí que un programa libre se denomine justamente gemelo digital. [6]

Respecto a las licencias privadas destinadas a la modelación, es importante aclarar que resultan indispensables y muy completas en cada aplicación, dependiendo del área del conocimiento, pero tal necesidad además de resultar excesivamente cara, demanda actualizaciones y mantenimientos constantes, aspectos que también generan gastos adicionales. En el área científica existen diferentes programas: Comsol Multiphysics, MatLab, Chemical calculators, Ansys, entre otros.

Con la evolución del software privado ha crecido también el formato libre que, como su nombre lo describe, es accesible al público especializado e incluso a cualquier usuario con conocimientos en informática. Los programas abiertos existen gracias a los programadores aficionados y compiten sin problema alguno con los sistemas operativos de paga.

El Proyecto GNU de la Free software Foundation considera que la principal cualidad de los códigos abiertos es la libertad que se ofrece a los usuarios para las siguientes acciones: "ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software". Es esta la principal ganancia entre los ejecutores de estos códigos y de quienes los usan.[1]

Se podría pensar que al ser abiertos son de dudosa calidad o que tienen deficiencias, pero no es así. Por el contrario, tienen cuatro grandes ventajas: 1. Son desarrollados por empresas privadas de prestigio. 2. Se pueden descargar en soportes como Windows o Mac, aunque algunos directamente en

Linux. 3. Estas empresas y centros de investigación testean y validan los programas antes de lanzarlos al público. 4. Siempre se mantienen actualizados, gracias a las aportaciones de usuarios y el financiamiento de clientes. [1]

Camilo Matías Díaz Alarcón, en su artículo "Software libre y gratuito para la simulación en Ciencias e Ingeniería" considera que esta cuarta ventaja es incluso sinónimo de superioridad frente a los códigos a la venta porque las "licencias de software comerciales limitan el uso total de núcleos de procesador para procesamiento en paralelo (CPU) y número de tarjetas de video GPU según el tipo y valor de la licencia adquirida, tema bastante alejado de la realidad de la capacidad de cómputo actual". [1]

Matías Díaz señala al menos dos empresas que han creado programas con código abierto: la empresa francesa Électricité de France (EDF) (creadora de Code Aster en Francia) y el centro de investigación avanzada NETL (National Energy Technology Laboratory (creadora de MFIX). [1] Existen, hasta ahora, una gama importante de software libre que pueden ser descargados desde internet. Camilo Matías hizo una importante recuperación de páginas que pone a disposición para la descarga de algunos de los programas más conocidos. [1] Describe a detalle cada uno de ellos, su funcionalidad, las áreas estratégicas en las que pueden resolver problemas, ventajas y desventajas. Aquí únicamente los mencionamos, pero se recomienda visitar la página para obtener una información más amplia:

1. CalculiX realiza modelados a partir de análisis de elementos finitos. [1]
2. OPENFOAM. Cada seis meses OpenCFD Ltd da a conocer actualizaciones que hace la comunidad virtual y que son patrocinadas por los clientes. [3]
3. FlexSim crea modelos a escala y promete contener uno de los paquetes más poderosos del modelado. Incluye sistemas autónomos y realidad ampliada. [4] "Evalúa las posibilidades futuras y luego informa al sistema del mundo real para tomar decisiones empresariales optimizadas". [6]
4. HSC Chemistry, uno de los más utilizados en el mundo, por su versatilidad. Alterna aspectos químicos, termodinámicos y mineralógicos; en este sentido multifactorial es innovador. [8]

5. Solid Edge. Se ofrece para el diseño de maquinaria industrial con ventajas como velocidad y modelado sencillo. [9]
6. Code_Aster analiza elementos finitos y simulación numérica; puede aplicarse en diferentes disciplinas, incluidas la acústica, sísmica y energía atómica.[10].
7. HELYX-OS combina licencias privadas y libres, dependiendo del uso pretendido, académico o industrial. [11].
8. LibreCAD, plataforma para diseños 2D, a partir de bibliotecas Qt4 y soportes en más de 30 idiomas. [12].
9. FreeCAD es un software para diseños mecánicos, modela sólidos, es recomendable para estudiantes de ingeniería. [13]

Al observar las descripciones de estas plataformas a fondo, será recurrente la invitación a consultarlos indicando al usuario, en la mayoría de los casos, que son de fácil manejo, pero en la realidad y al momento de ejecutar un prototipo para una circunstancia real, es posible que no sea tan sencillo.

Un punto clave de los códigos abiertos es que mantienen un aspecto *elitista*, por describirlo de algún modo, y consiste en que si no se cuenta con los conocimientos necesarios, no es posible acceder a su funcionamiento y mucho menos a los espectaculares resultados que pueden arrojar. Aquí es en donde el presente libro desea aportar una experiencia, en aras de que sirva como antecedente para quienes desean incursionar en la simulación abierta.

Como ya se ha indicado, en nuestro caso, enfocamos el interés en el área de la simulación. El software privado de esta categoría es, sin duda, el más caro en comparación con otros, debido a las diferentes áreas que puede atender, no solamente el rubro científico o académico, sino también la industria automotriz, farmacéutica, alimenticia o de la construcción.

En el sector automotriz es común la simulación, ya que se ocupa del desarrollo de prototipos mecánicos, eléctricos y mecánicos estructurales. Simular los elementos se refiere a que antes de crear el prototipo se debe generar el diseño del producto por medio de software enfocado a imágenes o figuras, y posteriormente aplicar la física a los elementos que actúan sobre el ente que será simulado. De esa forma se analizan los posibles resultados deseados o no deseados; de cualquier forma, esto nos da oportunidad de mejorar el producto o corregir el diseño antes de desarrollar un prototipo.

En la industria farmacéutica el software de simulación normalmente es usado en el diseño de contenedores o cajas para introducir los medicamentos. La idea de suponer objetos es asegurar que los diversos factores externos como humedad, temperatura, presión, etcétera, no alteren las propiedades químicas del producto y se tenga la seguridad de entregar un objeto confiable y con las características necesarias.

Una de las industrias que más recurre al software de simulación es la constructora, a raíz de que los proyectos arquitectónicos son imitados desde épocas inmemoriales, solo que ahora ha cambiado el soporte; ha transitado de los materiales manuales a escala al uso de tecnología virtual. Una vez que se tienen los resultados, se cuenta con los datos necesarios para llevar a cabo un proyecto de construcción sin riesgo alguno, ya que antes se analizan eventos probables que podrían afectar un inmueble, fenómenos naturales como inundaciones o temblores.

En el área científica, normalmente se pretende crear o innovar una tecnología. Por ejemplo, para fabricar los focos ahorradores, existió un desarrollo en el circuito y la forma del foco, a partir del prototipo generado por la simulación; con esta, los investigadores obtuvieron valores y parámetros tanto eléctricos como físicos que sirvieron para llegar al foco ahorrador. Seguramente ocuparon software privado para lograr el objeto comercializable, pero es probable que sean pocos los expertos que hayan conseguido lo mismo, a partir de software libre con la gran ventaja de la gratuidad.

En las descripciones breves de los diferentes programas abiertos se pudo observar que algunos se pueden combinar, dependiendo de la selección e intereses de los usuarios; de tal suerte que para efectos del presente libro se desarrollaron diferentes técnicas y métodos de simulación, como el de elemento finito o de volumen finito, cada uno con sus diversas características y por tanto aplicaciones en ejemplos básicos, como el de un cilindro, un tornillo o una lámpara de neón.

Para generar las geometrías de los entes a figurar se usaron los programas Salome – Meca, Code Saturne y ParaView. El primero ayudó a trazar la geometría; el segundo, a trabajar la física del ente y realizar los cálculos requeridos para la generación de reculados; por último, el tercero favoreció la interpretación de los resultados.

En la simulación existen varias etapas: generación de la geometría, concepción de la malla, y colocación de la física; a estos pasos se les conoce como preproceso, y al análisis e interpretación de los resultados se le conoce como posproceso. En las siguientes páginas se describe cada una de tales etapas, con el objeto de ejercitar y ofrecer una muestra, a partir de ejemplos sencillos, que el uso de código abierto puede resultar igual o mejor que el privado.

Resumen

En este libro ilustramo el paso a paso para diseñar objetos con software libre, particularmente desde los sistemas Salome Meca, Code Saturne y ParaView. Se intercalan ejemplos que propusimos simular y mostrar el funcionamiento de cada programa. Hemos enlazado las tres paqueterías como una forma de optimizar los recursos que los software ofrecen y adaptarlos a las necesidades que demandaron los proyectos que nos propusimos, en este caso una caja, un cilindro, un resorte, una abrazadera, un tornillo y una lámpara de luz neón.

Palabras clave: Simulación, Code Saturne, Salome Meca, ParaView, código abierto, programa abierto, geometría, malla, plasma frío.

Introducción

La multidisciplina es uno de los fenómenos que cobra mayor relevancia en la investigación científica, y es una aliada si de realizar diseños de modelación se trata. Para el desarrollo de simulación de seis objetos, a través de software libre, pretendimos conjuntar la manipulación de otros tres sistemas, fundamentalmente: Salome Meca, Code Saturne y ParaView. Todas las herramientas de modelación ofrecen cualidades importantes; sin embargo, para una aplicación más completa y profesional, se sugiere ocupar estos tres programas, dado que por la experiencia en la investigación realizada en el Laboratorio de Conversión de Energía en Tolouse, Francia, en 2013, se determinó que eran los más confiables y con una aproximación a resultados con un margen de error de 10 %.

Posteriormente se tuvo oportunidad de comprobar, en trabajos posteriores, esta eficiencia al interior del Laboratorio de Aplicación de Plasma del Instituto Nacional de Investigaciones Nucleares (ININ). Es gracias a estas particulares experiencias que nos atrevemos a recomendar la tríada para predecir escenarios dentro de la industria y la investigación.

El presente libro se divide en tres capítulos; el primero ofrece un panorama general de la instalación de Salome Meca y se presentan cinco diseños de objetos, a través de la geometría. En el segundo se da a conocer una guía para la instalación de Code Saturne y se desarrolla un objeto más, privilegiando la física. Finalmente, en el tercero se ofrece el camino para observar los resultados, con base en ParaView.

Hemos decidido hacer de esta propuesta escrita una guía dinámica para estudiantes y docentes que deseen incursionar en la modelación, dado que, en nuestra experiencia como investigadores, hemos encontrado una alta demanda en el medio académico para poner en marcha proyectos de diferentes tipos y distintas áreas del conocimiento.

Por lo anterior, en busca de su operatividad, la obra que el lector tiene en sus manos se conformó respetando los términos en idioma inglés que se van presentando a lo largo de los menús, y ofrecemos, por lo general, una traducción, y solo ocasionalmente describimos con mayor amplitud la funcionalidad de íconos y menús, dado que en gran parte de las muestras la única traducción ofrecerá la luz necesaria para un técnico observador.

A continuación, presentamos una breve descripción y explicamos la importancia de cada uno de los que seleccionamos para la simulación, mediante geometría. Salome Meca es un sofware de código abierto muy utilizado en el mundo, en las áreas científicas y tecnológicas, el cual proporciona una plataforma genérica para pre y posprocesamiento para la simulación numérica. Se basa en una arquitectura abierta y flexible hecha de componentes reutilizables. [13]

Salome Meca es una solución multiplataforma. Se distribuye como software de código abierto bajo los términos de la licencia GNU LGPL. Se puede descargar tanto el código fuente y los ejecutables de este sitio. [14] Puede ser utilizado como aplicación independiente para la generación del modelo CAD, su preparación para los cálculos numéricos y posprocesamiento de los resultados del cálculo. Sirve como plataforma para la integración de los códigos numéricos de terceros externos para producir una nueva aplicación en la gestión del ciclo de vida completo de los modelos CAD. [14]

El usuario puede ocupar los malladores incluidos, libres o de código cerrado, definir las condiciones de contorno, ejecutar el programa de solución, visualizar el resultado y analizar los datos. Salome puede hacer grandes simulaciones numéricas. En este contexto, el posprocesador más potente es llamado ParaView y lo integramos en la plataforma de Salome para beneficiarnos de sus capacidades y poder visualizar modelos a gran tamaño. Las plataformas sobre las que se puede descargar pueden ser Linux, Windows o Mac, con ciertas limitantes, dependiendo de la plataforma que se ocupe; [14] sin embargo, es recomendable hacerlo sobre Linux.

Salome cuenta con los módulos de Geometría (GEOM), de Mallado,y posprocesamiento ParaView. En el primero se establece la geometría del problema a resolver, mediante Open Source Cascade, o bien esa geometría se puede importar desde los formatos IGES, STEP, BREP y Python. Después viene la preparación de la malla y el análisis numérico. [14]

En el segundo módulo de mallado se pueden crear mallas en 1D, 2D y 3D, a través de algoritmos propios o externos que forman parte de la interfaz de Salome, estos últimos pueden ser de código abierto (NETGEN 4.9), o bien de paga (GHS3D, GHS3D parallel, BLSURF, Hexotic). Por obvias razones, si se elige usar un código comercial, este deberá adquirir antes la licencia. [14]

Otro software que utilizamos para el ejercicio, en particular, de la lámpara de neón fue Code Saturne, y el principal en el desarrollo de este objeto, porque en él se importó la geometría discretizada generada anteriormente en Salome Meca para colocar la física en el sistema en Code Saturne. La física son las variables que involucran al sistema, como ejemplo: la temperatura, la presión, el voltaje, la velocidad, etcétera. Para la colocación de la física en la geometría, se ocupó la programación Python que nos permitió, por medio de coordenadas y de instrucciones, ir colocando las diferentes variables de la física a utilizar en la simulación de la lámpara de neón.

Al igual que con Salome Meca, para usar esta aplicación fue necesario instalar el software en la computadora personal (PC). Al momento de la instalación se descargó el programa de la página oficial de Code Saturne. Había dos maneras de realizarla; pero en este libro se muestra la vía más simple, donde lo más importante para poder realizar la instalación, era estar conectado a internet, debido a que el software descargaría e instalaría las librerías fundamentales para la conformación completa de Code Saturne en forma automática.

Con el tercer y último software denominado ParaView, fue posible visualizar los resultados proporcionados por Code Saturne, el cual genera una carpeta que contiene los archivos de los resultados obtenidos de los cálculos de la simulación numérica. Code Saturne no tiene la capacidad para observar e interpretar estos resultados, lo cual sí sucede con ParaView que también permite modificar las escalas de colores para obtener una representación más exacta de los hallazgos. Para el uso de ParaView no es necesario

instalarlo, solo se requiere descargar el software de la página oficial y ejecutar el programa en una PC, sin problema.

Así también, en el posprocesamiento con ParaView se desarrolló el análisis numérico de la simulación realizada, mediante técnicas cualitativas y cuantitativas. ParaView es un sistema con capacidad para revisar datos extensos, por lo que se puede usar en superordenadores. [14]

Es evidente entonces que se pueden realizar combinaciones de licencias, pero las decisiones siempre dependerán de que se adapten a las necesidades de la industria o proyecto que se esté realizando. En el caso de los distintos objetos, incluida la lámpara de neón que se desarrolla en la presente contribución, hemos combinado los componentes antes citados, a razón de su operabilidad y viabilidad.

Instalación de *Salome Meca*

En el presente capítulo y en los posteriores mostraremos paso a paso cómo debemos instalar Salome Meca, Code Saturne y ParaView, además se intercalarán los momentos en los que estos van efectuando los distintos objetos que nos propusimos diseñar, en este caso una caja, un cilindro, un resorte, una abrazadera, un tornillo y una lámpara de luz neón. Hemos entrelazado los tres programas para dar a conocer las pautas que seguimos al momento de ejecutar el presente ejercicio. Inicialmente presentamos la instalación del programa Salome Meca al tratarse de la herramienta que ayudará a elaborar la geometría del objeto.

A lo largo de las siguientes páginas, el lector encontrará descripciones ágiles porque se han traducido los términos en inglés más importantes que aparecen en cada ícono; se muestran los menús acompañados de imágenes de las ventanas que se van abriendo al manipular cada uno de los tres software, a fin de que los lectores puedan contar con una brújula aproximada.

Decidimos, asimismo, ocupar un lenguaje breve y técnico tanto para comodidad de quienes intenten aplicar el ejercicio, o bien para otros lectores que deseen basarse en este modelo y puedan generar un gemelo digital distinto.

A continuación, mostramos la secuencia para instalar Salome-meca. Inicialmente descargaremos la versión Salome-Meca 2016 desde el sitio oficial, como se muestra en la figura 1.

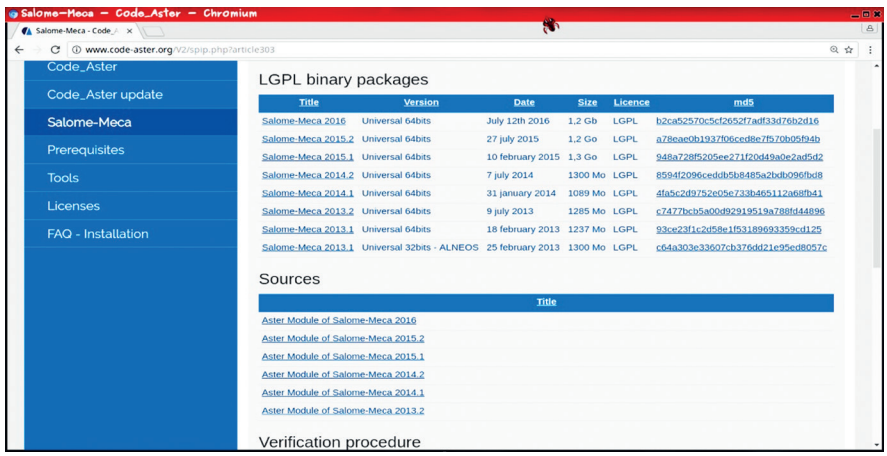


Figura 1. *Descarga Salome Meca 2016*

Primero abriremos nuestra terminal: Inicio>terminal o Ctrl>alt+t = terminal. Accederemos al directorio donde tenemos descargado nuestro programa, guardado en el directorio SalomeMeca2016 al cual accederemos como se observa en la figura 2.

Figura 2. *Programa descargado*

Encontraremos el archivo comprimido SALOME-MECA-2016-LGPL-l.tgz que descomprimiremos, según aparece en la figura 3.

Figura 3. *Archivo descomprimido*

Al descomprimir tendremos un archivo con extensión .run, el cual ejecutaremos con nuestra terminal ./SMECA_V2016_LGPL.run para que continúe con la instalación mostrada en la figura 4.

Figura 4. *Agregando la terminal*

Nos dará la ruta en automático en la que se guardará [*default=/home/Fernando/salome_meca]*; ahí se puede modificar o dejar que te la proporcione en automático, tal como se muestra en la figura 5.

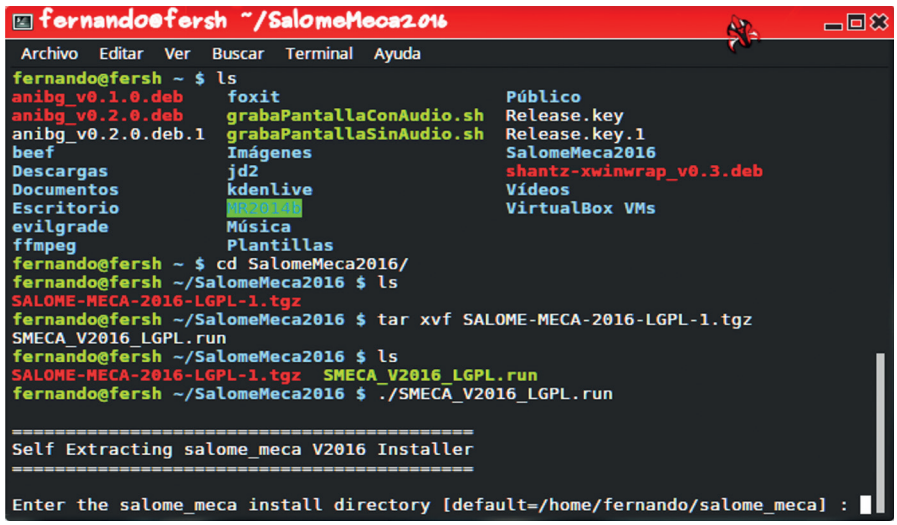


Figura 5. *Guardado de la ruta en automático*

Después instalamos los paquetes, según la figura 6.

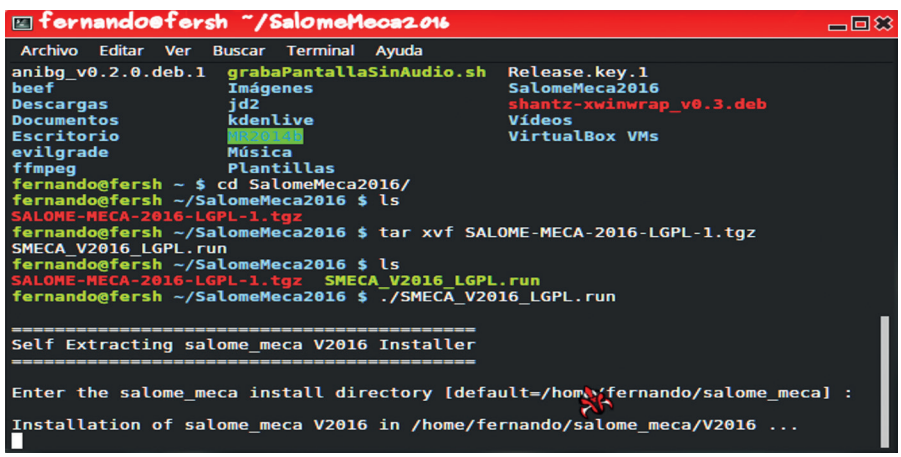


Figura 6. *Instalación de paquetes*

Si regresamos un directorio veremos una carpeta del programa ya instalado: **salome_meca**, en la cual están los archivos necesarios para iniciarlo como se visualiza en la figura 7.

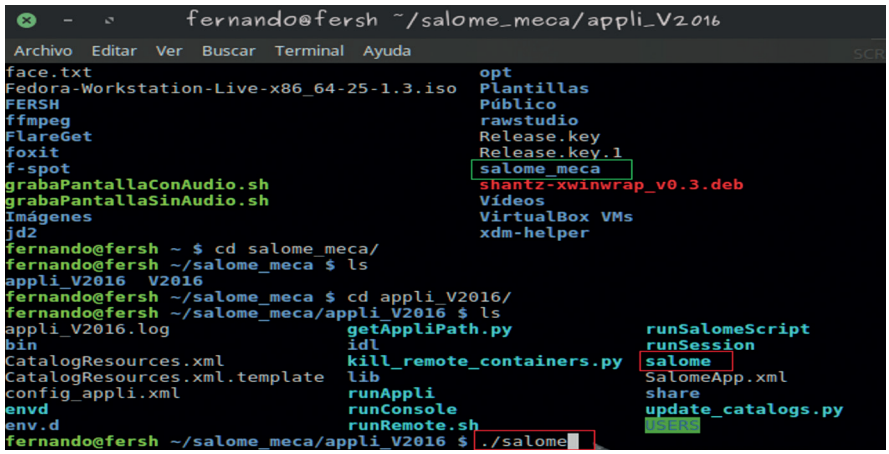


Figura 7. *Carpeta del programa instalado*

Inicio de *Salome Meca*

A continuación, presentamos la interfaz principal del software Salome Meca. Sobre la ubicación del puntero tenemos las diferentes áreas de trabajo con las que este software cuenta; para este caso nos enfocaremos en Geometry y Mesh, íconos que vemos en la figura 8.

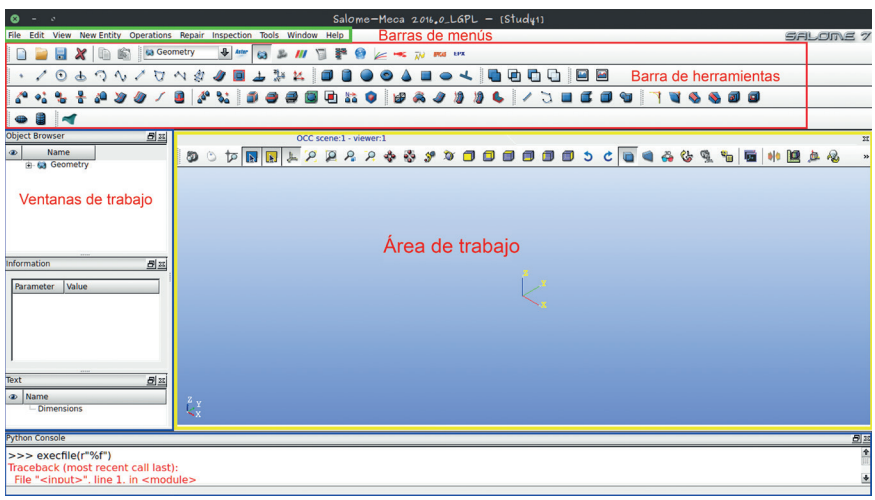


Figura 8. *Interfaz principal Salome Meca*

En la figura 9 tenemos la pantalla principal de Salome Meca Geometry, donde aparece el recuadro verde y sobre este encontramos la barra de menú; en ella podemos acceder a todas las herramientas y utilidades que este sof-

tware nos ofrece en el recuadro rojo. En la barra de herramientas nos encontramos con accesos rápidos a algunas de las herramientas que utilizaremos para nuestros proyectos; se puede modificar a su comodidad, según veremos más adelante. En el recuadro azul destacan las ventanas de trabajo desde las que podemos observar Object Browser, es decir todos aquellos procesos o pasos que podemos renombrar dando doble clic sobre el nombre del objeto o categoría que queramos cambiar; después tenemos Information para ver los datos de cada uno de los objetos y pasos que realicemos, enseguida observamos Python Console con el que podemos hacer nuestros proyectos con ayuda del lenguaje de programación que muestra sobre el recuadro amarillo el área de trabajo, interfaz en la que están plasmados nuestros objetos y resultados de cada operación. Ahora describimos cada uno de los elementos de importancia para nuestro cometido.

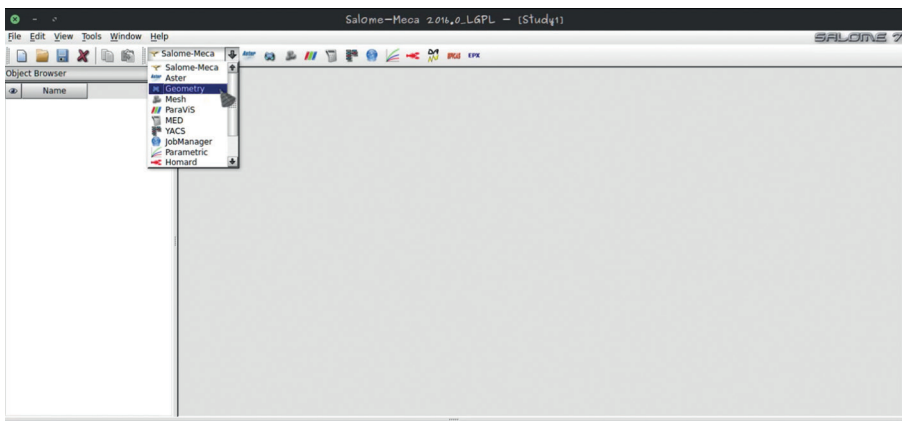


Figura 9. *Página principal Salome Meca*

Elementos de trabajo


Dump View: crea una imagen con los formatos *.bmp,*.png,*.jpg,*.eps,*.ps


Interaction Style Switch: crea una interacción del puntero para rotar los elementos.


Zooming Style Switch: crea una interacción de un zoom con el puntero.


Enable/Disable Preselection: activa y desactiva la preselección cuando el cursor se posa sobre el objeto (sobre el objeto sobresale un color).


Enable/Disable Selection: activa y desactiva la selección de los objetos.


Show/Hide Trihedron: muestra/ oculta el triedro (las coordenadas del eje X, Y, Z).


Fit All: enfoca el objeto.


Fit Area: resalta un área del objeto.


Fit Selection: resalta al objeto seleccionado.


Zoom: agranda o minimiza el objeto.


Panning: mueve el objeto.


Global Panning: centra el objeto en la pantalla.


Rotation: rota el objeto.


-OX: vista sobre el eje X positivo (izquierda).


+OX: vista sobre el eje X negativo (derecha)


-OZ: vista sobre el eje Z positivo (superior)


+OZ: vista sobre el eje Z negativo (inferior)


+OY: vista sobre el eje Y negativo (trasera)


-OZ: vista sobre el eje Y positivo (delantera)


Rotate Counterclockwise: rotación al lado contrario de las manecillas del reloj.


Rotate Clockwise: rotación al sentido de las manecillas del reloj.


Reset: restauración de la vista.


Memorize View: memoriza una vista que deseemos.


Restore View: recupera una vista antes memorizada.


Clone View: clona una ventana, creando dos que podemos cerrar después.

Figura 10. *Elementos de trabajo*


Scaling: escala el objeto definido por cada eje.


Graduated Axes: muestra información de las dimensiones del objeto de manera gráfica. Ver figura 12.

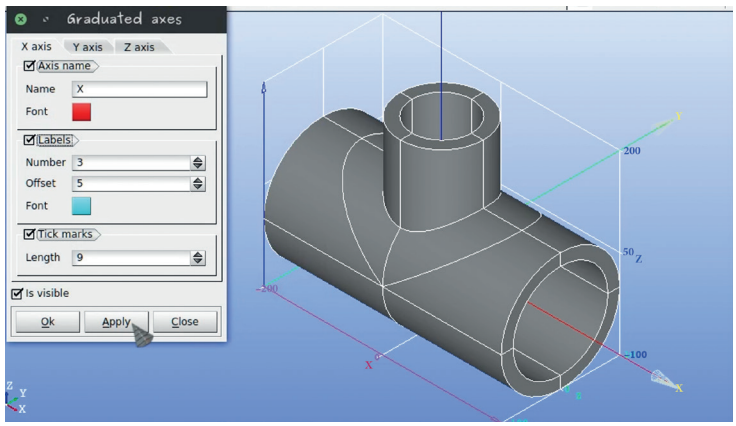


Figura 11. *Graduated Axes*

Minimize: nos crea una subvista de diferente manera ya predeterminada. Ver figura 12.

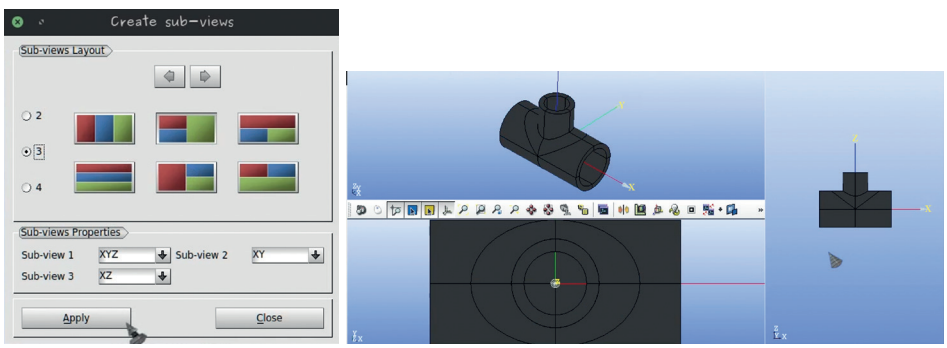


Figura 12. *Create Sub-Views*

Maximize: maximiza la vista.

Return to 3D: regresa la vista 3D.

Environment texture: podemos cambiar la textura de nuestros objetos. Figura 13.

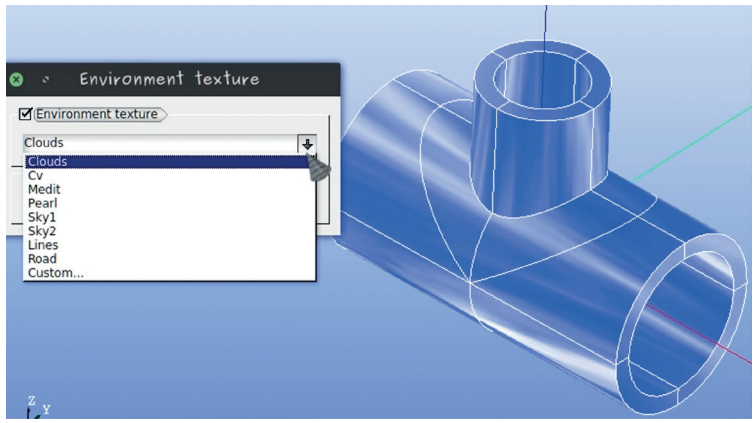


Figura 13. *Environment Texture*

Podemos cambiar la textura del fondo del área de trabajo como vemos en la figura 14. Damos clic derecho en alguna parte del área de trabajo y elegimos Change Backgroud, ahí podemos elegir los colores deseados.

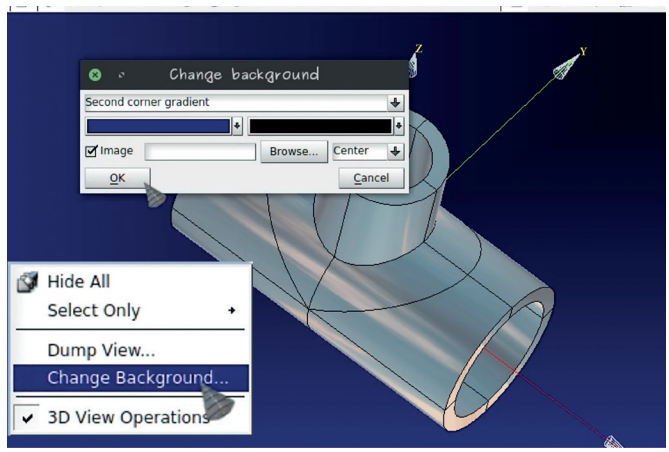


Figura 14. *Cambiar textura de fondo*

Menú File

Con las Properties de **Menú File** podemos elegir el tipo de unidad en la que deseas trabajar cm, mm, m, km o inch (pulgadas). Ver figura 15.

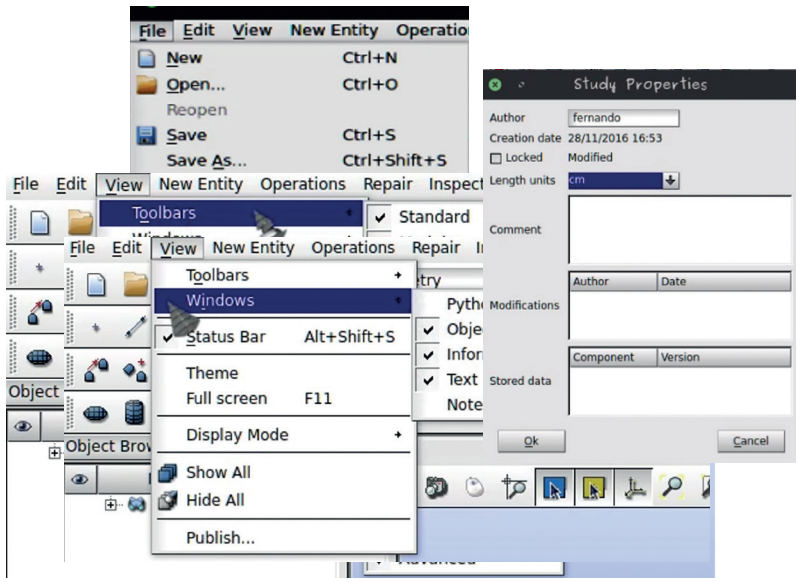


Figura 15. *Menú file*

Menú View

Esta opción contiene una gama importante de herramientas y utilidades. Veremos algunas de ellas que son indispensables:

1. **Toolbars.** En la barra de herramientas podemos activar o desactivar los diferentes menús que ofrece el programa.
2. **Windows.** Aquellas ventanas que hemos visto anteriormente como la de Object Browser, se pueden activar y desactivar según nuestra preferencia.
3. **Theme.** Podemos elegir diferentes temas para nuestra interfaz del sistema.
4. **Full screen.** Haremos que el área de trabajo esté en pantalla completa.

Display Mode

Es el modo en que podemos visualizar nuestros objetos con siete alternativas para observar diferentes enfoques del objeto que estamos diseñando.

1. **Wireframe** permite observar únicamente los bordes de los objetos. Ver figura 16.

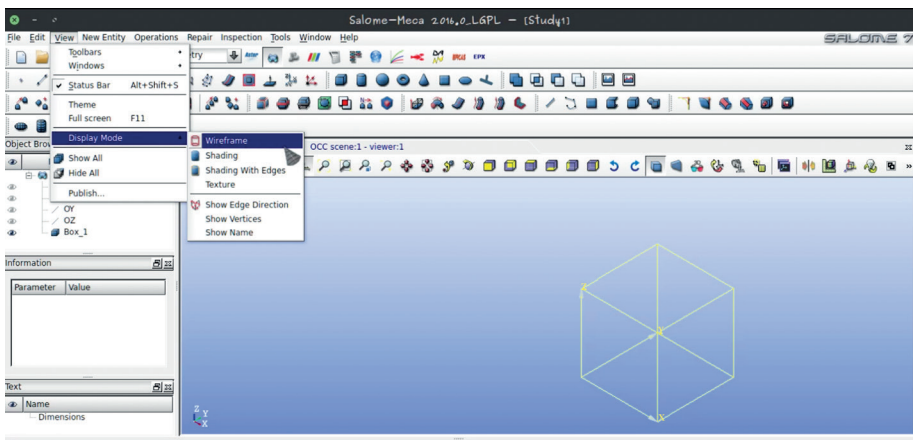


Figura 16. *Wireframe*

2. **Shading** activa la visualización de los objetos, pero sin los bordes, con un color sólido. Figura 17.

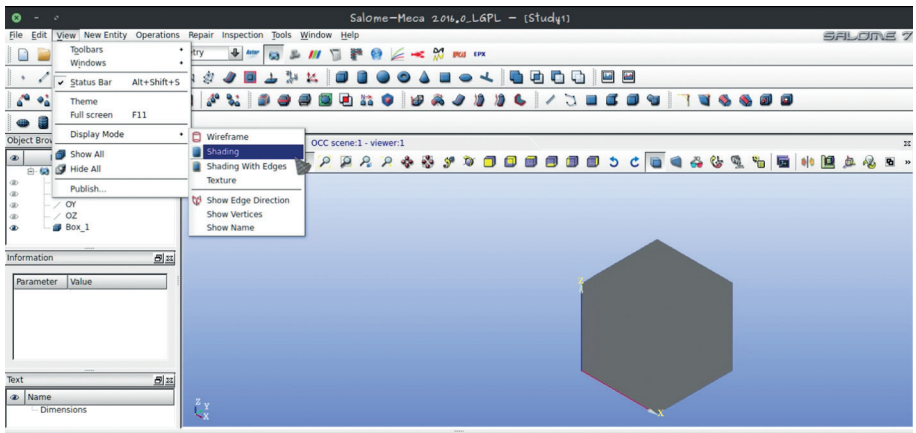


Figura 17. *Shading*

3. **Shading with Edges** es para ver los objetos, pero con los bordes.

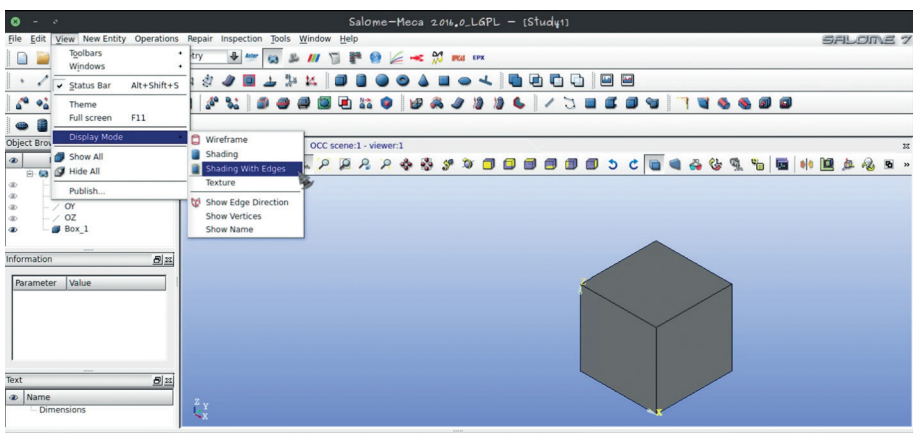


Figura 18. *Shading With Edges*

4. **Texture** nos mostrará los objetos con una textura, en automático. Figura 19.

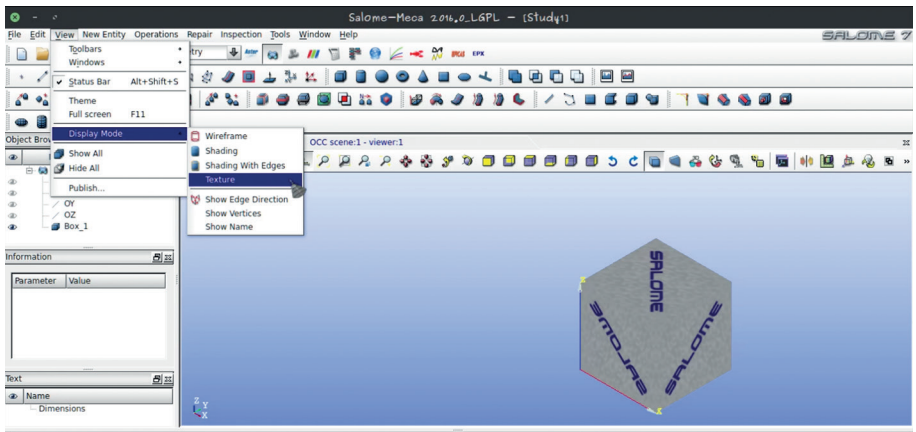


Figura 19. *Texture*

5. **Hide Edge Direction** nos da la opción de ver los bordes con la dirección que lleva, indicando con una flecha. Figura 20.

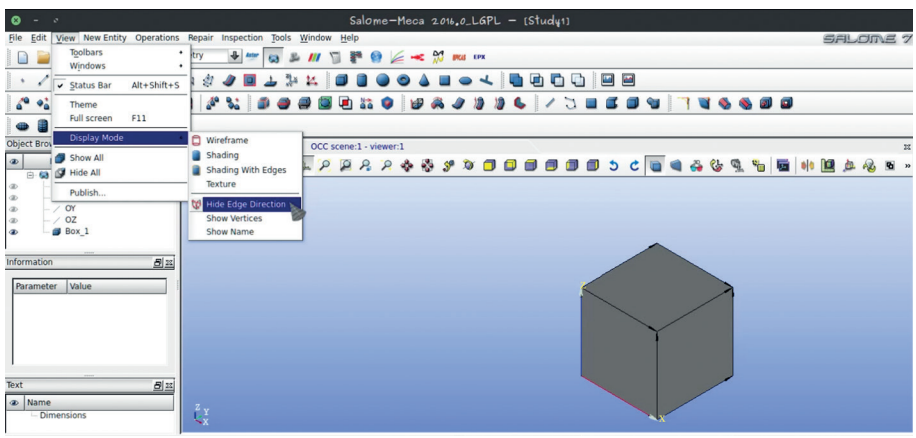


Figura 20. *Hide Edge Direction*

6. **Hide Vertices** activa o desactiva para que podamos observar los vértices de nuestros objetos. Figura 21.

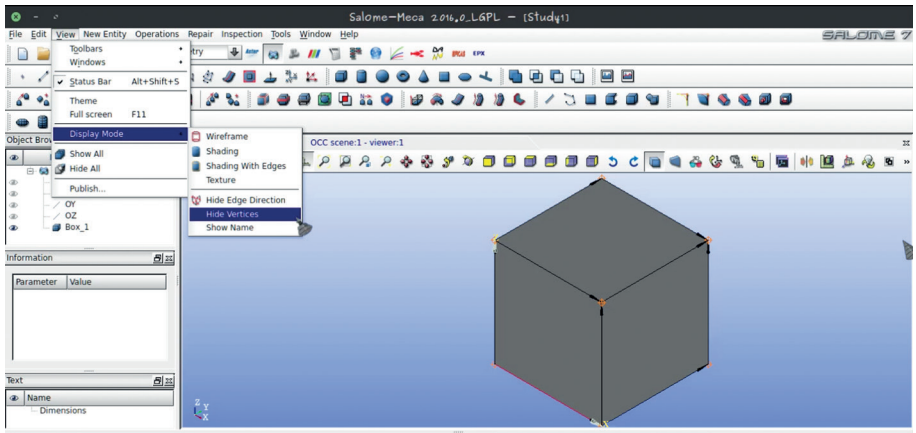


Figura 21. *Hide Vertices*

7. **Show Name** activa o desactiva que podamos ver el nombre de los objetos. Figura 22.

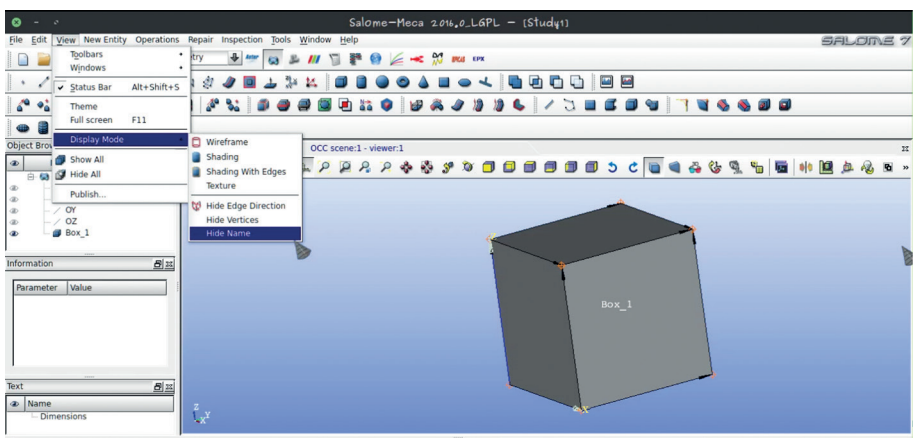


Figura 22. *Show Name*

Object Browser

Ahora veremos algunas utilidades para visualizar nuestros objetos sobre la ventana **Object Browser**, algunas de ellas no están en el menú View, pero forman parte del botón secundario del mouse.

1. **Show All.** Podemos activar todos los objetos para verlos en nuestra área de trabajo. Figura 23.

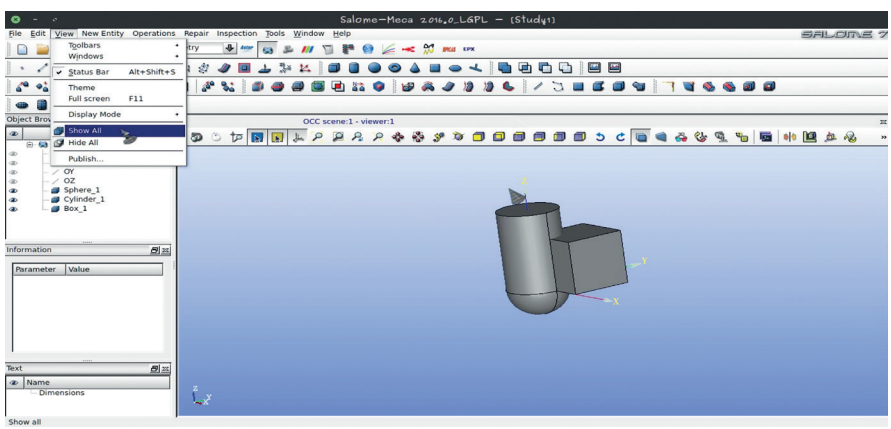


Figura 23. *Show All*

2. **Hide All.** Desactiva la visualización de todos los objetos en el área de trabajo. Figura 24.

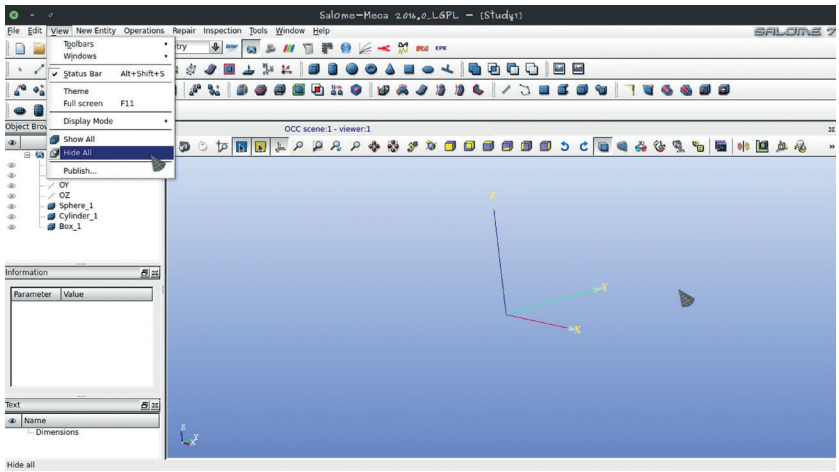


Figura 24. *Hide All*

3. **Hide.** Si nos colocamos sobre el nombre de algún objeto en la ventana Object Browser, damos un clic con el botón secundario y elegimos Hide para desactivar el objeto, lo cual no será visualizado. Figura 25.

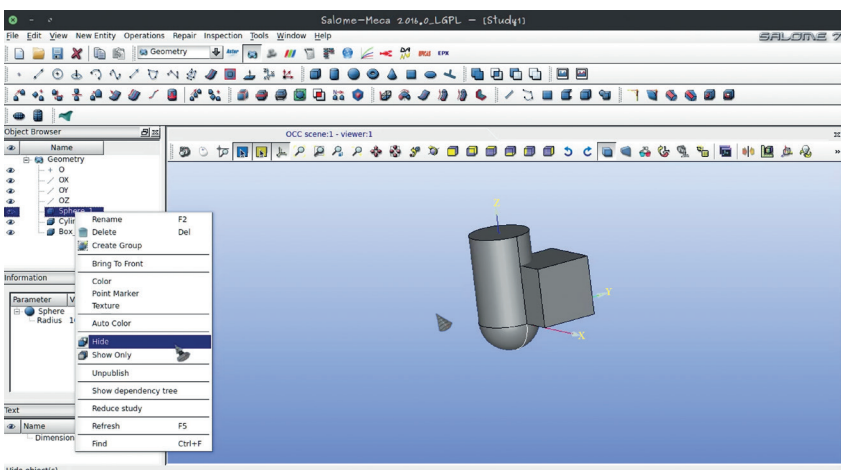


Figura 25. *Hide*

4. **Show.** Si tenemos algún objeto oculto, Show nos permitirá activar el objeto para poder visualizarlo en el área de trabajo. Figura 26.

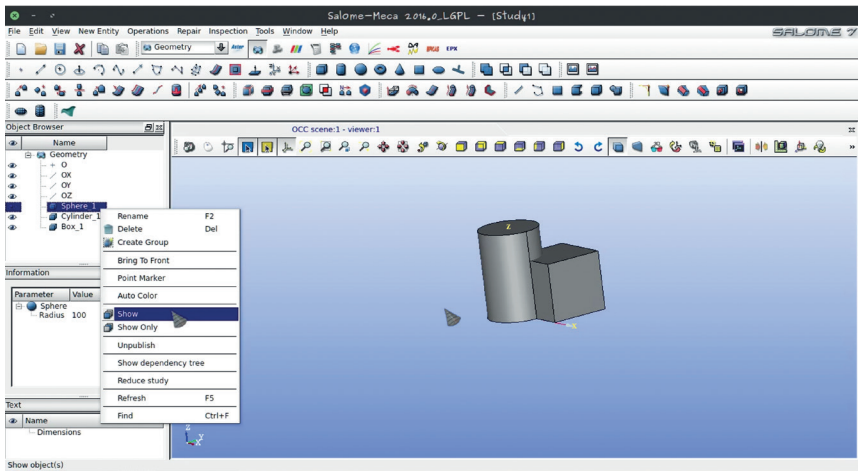


Figura 26. *Show*

5. **Show Only.** Esta opción nos permite que solo ese objeto seleccionado se visualice desactivando el resto de la forma. Figura 27.

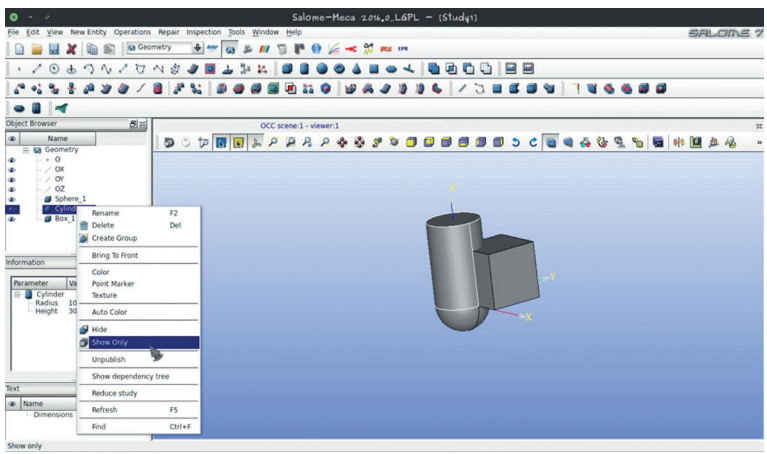


Figura 27. *Show Only*

De igual manera podemos activar y desactivar como se muestra en la figura 28 cualquier objeto dando un clic sobre el ojo en la ventana Object Browser. También podemos eliminar objetos seleccionando aquel objeto y oprimiendo la tecla Supr o Del o clic derecho sobre el elemento en la opción Delete y aceptamos la eliminación.

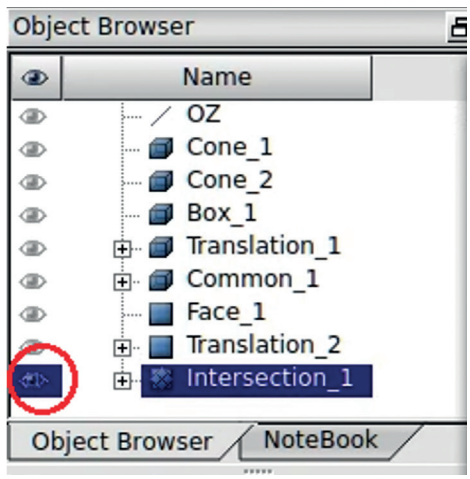


Figura 28. *Activar y desactivar objetos*

1. **Ctrl + clic.** Este comando sirve para seleccionar los elementos que deseemos, lo podemos usar tanto en la ventana Object Browser como en el área de trabajo. Figura 29.

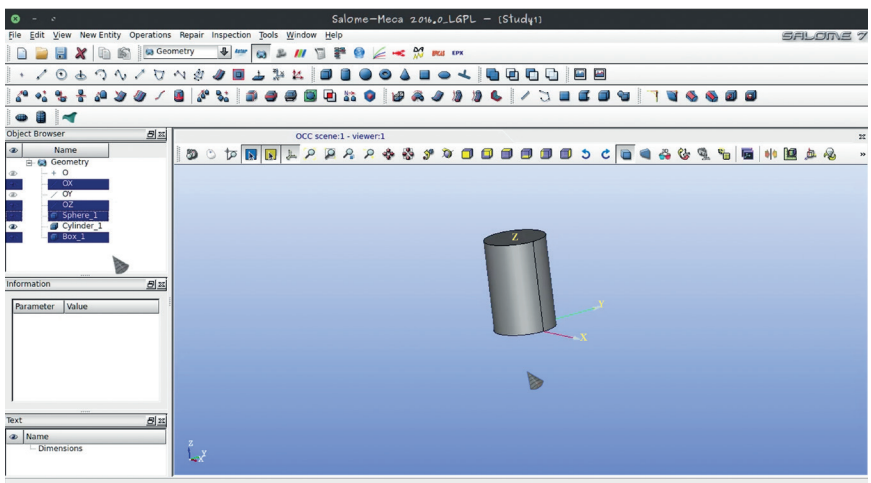


Figura 29. *Selección de elementos*

2. **Shift + clic.** Con esta combinación podemos seleccionar desde un punto de elementos a otro, seleccionando todos aquellos que estén dentro del rango elegido. Figura 30.

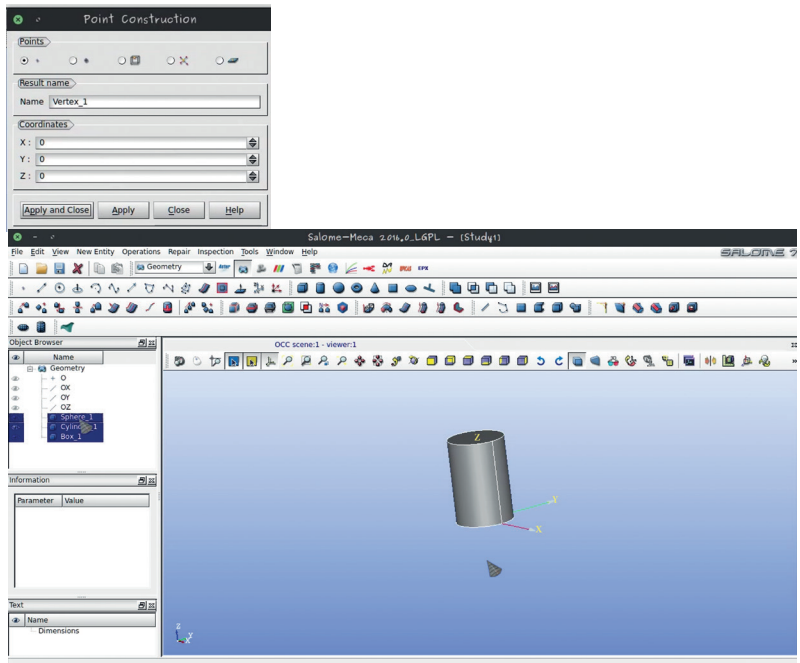


Figura 30. *Selección de elementos dentro del rango.*

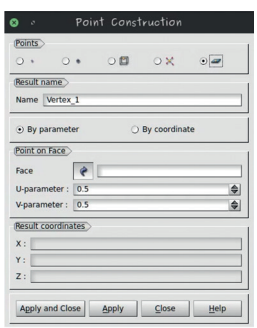
New Entity

Ahora veremos herramientas del menú New Entity y explicaremos brevemente su función. Para ello nos dirigimos al menú **New Entity > Basic** donde podremos observar nueve alternativas: Point, Line, Circle, Ellipse, Arc, Curve, 2D Sketch, Vector y Local Cordinate System.

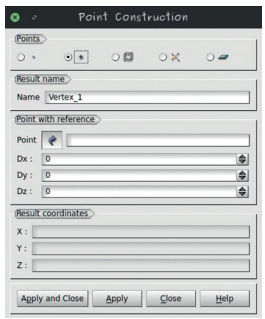
1. Point (Punto)

Se ocupa para definir un punto estableciendo sus coordenadas X, Y y Z, como se puede advertir en el conjunto de cuatro ventanas que conforman la figura 31.

Argumentos: nombre del objeto y valores en las coordenadas X, Y y Z.



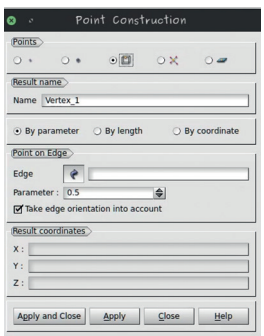
Definir un punto por una referencia a otro punto y el desplazamiento sobre las coordenadas.
Argumentos: nombre del objeto, punto de referencia, valor de las coordenadas que definen la posición de este punto respecto a la referencia.



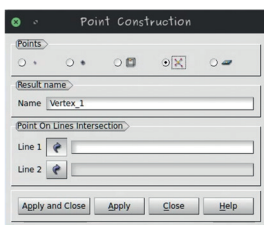
Definir un punto por un borde y un parámetro que indica su posición en el borde, que va de 0,0 a 1,0.
Argumentos: nombre del objeto, elegimos el borde, damos valor al parámetro que define la posición del punto en el borde, la casilla indica si es necesario tener en cuenta la orientación del borde.

Punto en un borde y una longitud. La longitud define la posición del punto en el borde dado.

Punto por coordenadas 3D del punto proyectado en el borde, dado para producir el punto resultante.



Definir un punto por intersección de dos líneas. Si hay varias intersecciones, se creará un compuesto de puntos.
Argumentos: nombre de objeto, elige línea 1, elige línea 2.



Definir un punto localizado en una cara. La posición del punto se puede definir de dos maneras:
Opción 1. Por dos Parámetros: U y V, comprendidos entre 0,0 y 1,0.
Argumentos: nombre del objeto, Face a elegir, dos parámetros U y V que definen la posición del punto en la cara.
Opción 2. Por coordenadas sobre la cara dada.
Argumentos: nombre del objeto, Face a elegir, valor de coordenadas X, Y, Z.

Figura 31. *Cuatro ventanas para observar la definición de puntos y líneas*

2. Line (Línea)

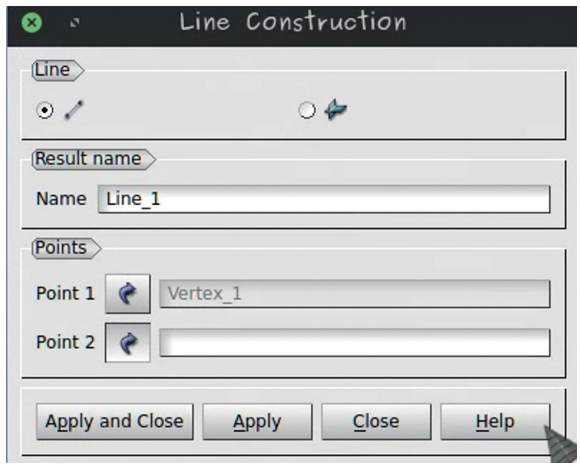


Figura 32. *Definir una línea por intersección*

Opción 1. Puede definir una línea por punto 1 y punto 2.
Argumentos: nombre del objeto y sobre Point 1 y 2 elegir los vértices. Figura 33.

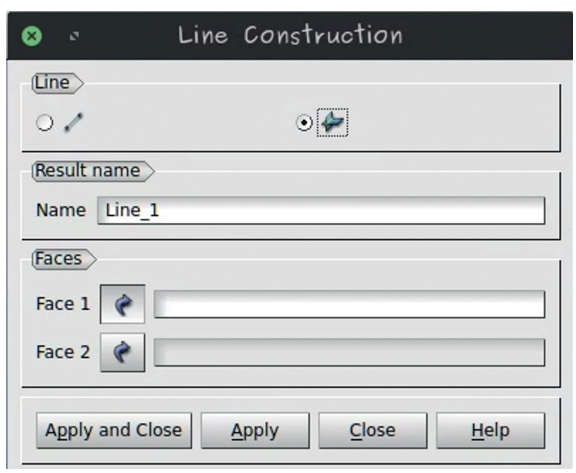


Figura 33. *Definir una línea por punto*

Opción 2. Puede definir una línea como intersección de Plane 1 y Plane 2.
Argumentos: nombre del objeto y elegir las dos caras (Faces). Figura 32.

3. Circle (Círculo)

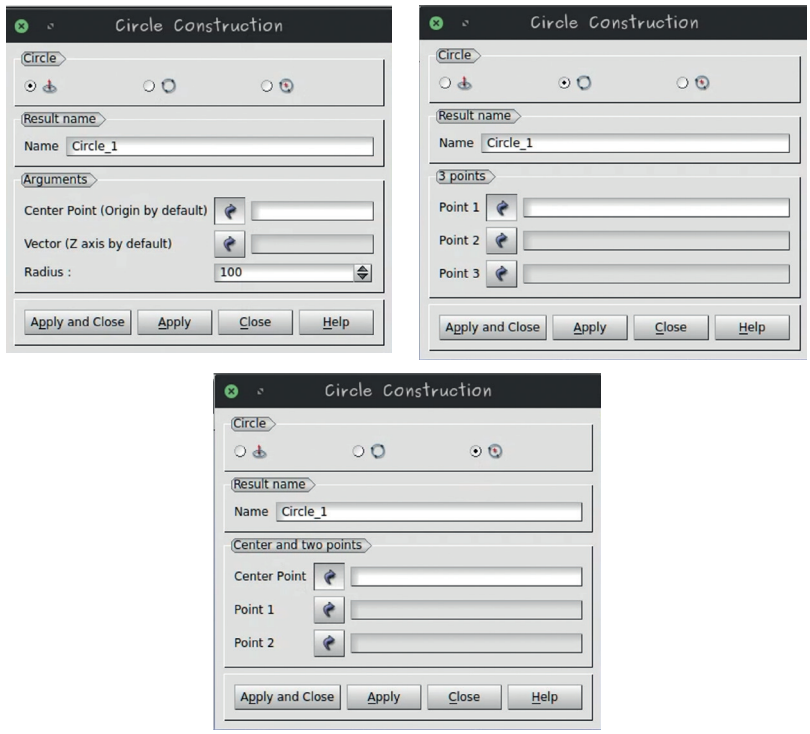


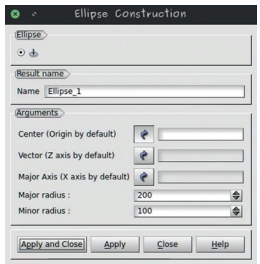
Figura 34. *Circle*

Opción 1. Definir un círculo por un punto central, un vector que da el círculo y un radio.
Argumentos: nombre del objeto; en Center Point; elegir vértice; sobre vector elegir borde (para la dirección), más la dimensión del radio.

Opción 2. Definir un círculo por tres puntos.
Argumentos: nombre del objeto, definir los tres puntos que formarán el círculo.

Opción 3. Definir un círculo por un punto central y dos puntos.
Argumentos: nombre del objeto, define puntos 1, 2, y 3, donde Point 1 es el centro del círculo, la distancia entre punto 1; Point 2 es el radio del círculo y Point 3 define el plano.

4. Ellipse (Elipse)



Definir una elipse por su punto central, un vector que da su posición normal, vector o eje que especifica la dirección del eje mayor de la elipse y sus radios.

Argumentos: nombre del objeto, Center elige el vértice (para el centro), sobre vector elige borde (para la dirección normal), sobre Major Axis elige borde (para la dirección del eje principal), valor de los radios de la elipse.

Figura 35. *Ellipse*

5. Arc (Arco)

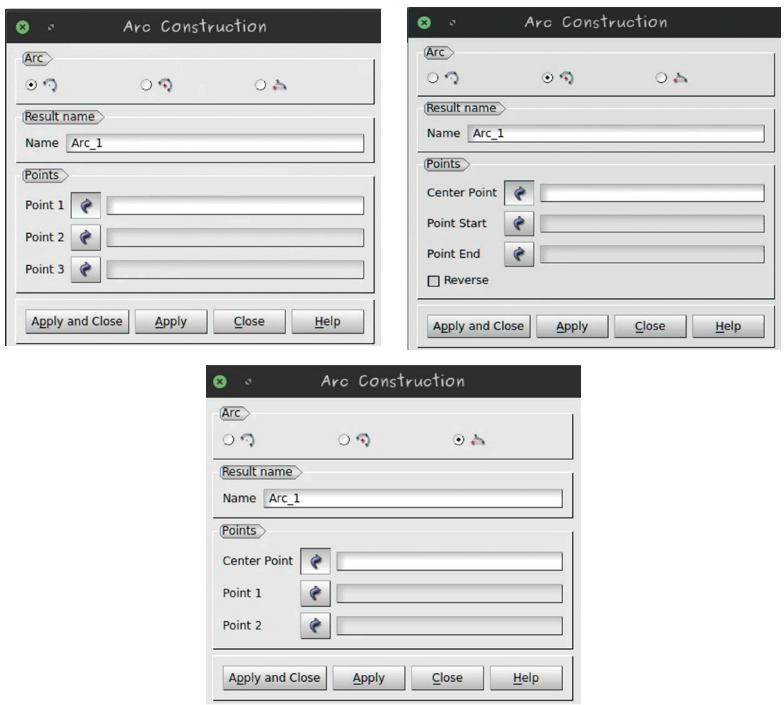


Figura 36. *Arc*

Opción 1. Definir un arco por tres puntos, donde punto 1 es el punto de partida, punto 2 es un punto medio del arco y punto 3 es el punto final del arco.

Opción 2. Definir un arco por centro, Inicio y Puntos finales. El arco se construye desde el punto de inicio hasta el punto final. El radio del arco se define por la distancia entre el punto central y el punto de inicio. El punto final define el ángulo del arco.

Opción 3. Es posible crear un Arco de Elipse, también se hace por tres puntos. El arco se construye sobre la elipse que se encuentra en el plano definido por los tres puntos.

6. Curve (Curva)

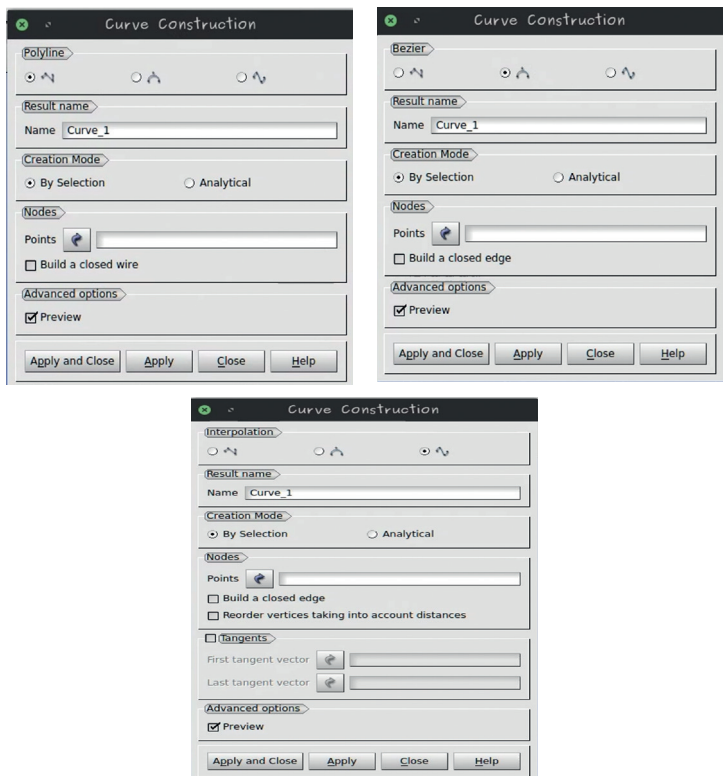


Figura 37. *Curve*

Hay tres opciones de curvas: polilínea, Besier o B-spline (interpolada). La curva se define por una lista de puntos a través de los cuales pasa. Existen dos maneras de definir estos puntos:

- Selection. Selección manual de los puntos en el Explorador de objetos o Visor 3D.
- Analytical. Analítica paramétrica de los puntos.

Opción 1. Polilínea es una serie conectada de segmentos de línea. Se puede definir mediante los siguientes parámetros:

- Points. Por lo menos 2 puntos que servirán como nodos en la curva.
- Build a closed wire. Permite crear la curva como un borde cerrado.

Opción 2 Bezier. Es una curva completamente contenida en un casco convexo de sus puntos de control. Se puede definir mediante los siguientes parámetros:

- Build a closed Edge. Permite crear la curva como borde cerrado.
- Point. Por lo menos 2 puntos utilizados para aproximar la curva.

Opción 3 B-spline. Es una unión de segmentos de curva definidos en cada intervalo de nodos. Se puede definir mediante los siguientes parámetros para la Analytical:

- La ecuación X (t), la ecuación Y (t), la ecuación Z (t) son expresiones de Python para las coordenadas X, Y y Z de los puntos básicos de la curva.
- Min t, Max t son valores mínimo y máximo del parámetro t.
- Number of Step son pasos del parámetro t.

7. 2D Sketch. Permite dibujar formas 2D arbitrarias. Figura 38.

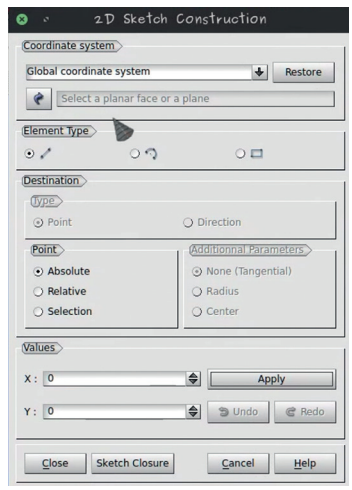


Figura 38. *2D Sketch*

Permite crear bocetos de tipo línea, arcos o rectángulos, elegir el tipo de elemento a crear, elegir tipo de Destination puede ser Point o Direction, elegir el punto de inicio, el cual se puede realizar de tres formas:

- Coordenada Absolute (absoluta). Ingrese los valores de X y Y, haga clic en el botón Aplicar.
- Coordenadas Relative (relativa). Ingrese los valores DX y DY y haga clic en el botón Aplicar.
- Selection (selección). Seleccione un punto en el navegador de objetos y haga clic en el botón Aplicar.

Como alternativa a la definición por Direction, tenemos que puede ser:

- Definido por un ángulo (Angle) al segmento anterior.
- Perpendicular al segmento anterior (es decir, el ángulo es de 90 grados).
- Tangente (Tangent) al segmento anterior (es decir, el ángulo es de 0 grados).
- Definido por las coordenadas del vector VX-VY.

La distancia se puede definir:

- Por el segmento absoluto Longitud (Length).
- Por DX - la longitud de la proyección de segmento en el eje X.
- Por DY - la longitud de la proyección de segmento en el eje Y.

8. Vector

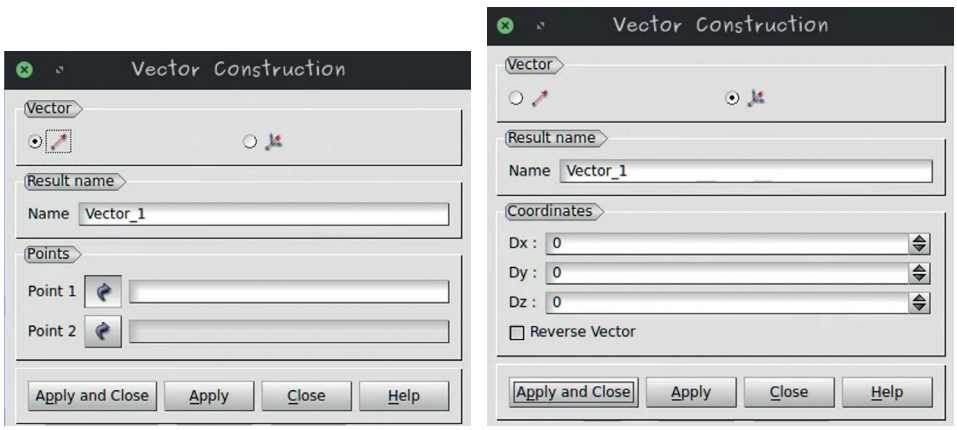


Figura 39. *Vector*

Opción 1. Definir un vector por sus puntos de inicio y final.
Argumentos: nombre del objeto, elegir los vértices en las alternativas Point 1 y 2.

Opción 2. Definir un vector comenzando en el origen de las coordenadas por su punto final.
Argumentos: nombre del objeto, valores a las direcciones Dx, Dy y Dz.

La casilla Reverse Vector cambia la dirección contraria de la dirección del objeto.

9. Local Coordinate System

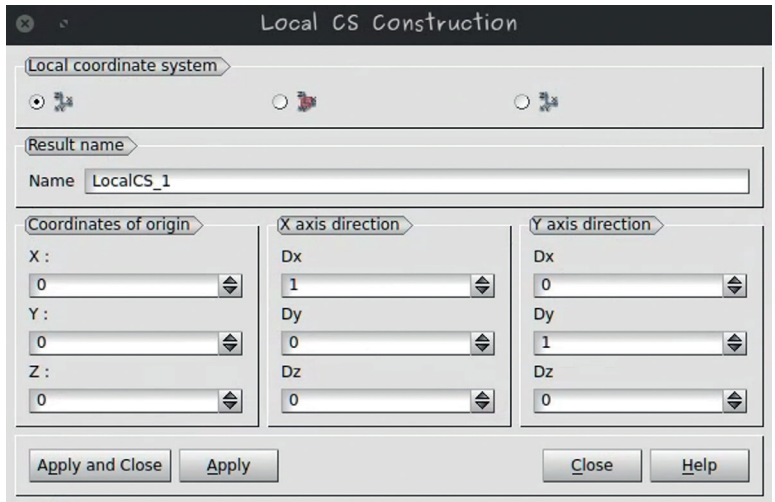


Figura 40. *Local Coordinate System*

Opción 1. Definir los valores de coordenadas X, Y y Z de origen y las direcciones de los ejes X y Y directamente en el menú.
Argumentos: nombre, coordenadas de origen, dirección del eje X, dirección del eje Y.

Opción 2. Puede seleccionar cualquier objeto en el navegador de objetos, en este caso las coordenadas de origen y dirección de los ejes del LCS son calculadas basándose en el objeto seleccionado.
Argumentos: nombre y objeto de referencia.

Opción 3. Permite definir las coordenadas de origen por un punto y direcciones de ejes por una línea o un vector.
Argumentos: nombre, punto de origen (Point), dirección del eje X y dirección del eje Y.

New Entity > Primitives

Enseguida nos dirigimos al menú **New Entity** > **Primitives** en donde podremos observar ocho funciones, que son: Box, Cylinder, Sphere, Torus, Cone, Rectangle, Disk y Pipe Tshape.

1. Box (cubo o caja)

Para el diseño del modelo que requerimos tenemos que elegir entre dos alternativas:

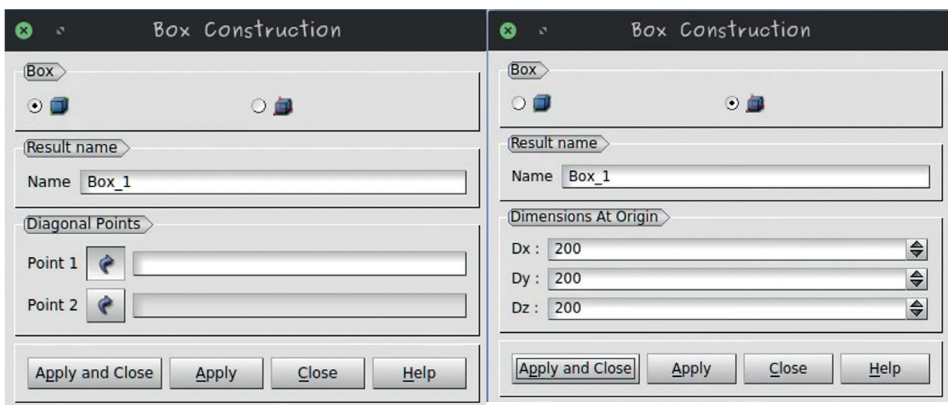


Figura 41. *Box*

Opción 1. Definir un cubo por dos vértices especificados.
Argumentos: nombre del objeto, dos vértices (sobre Point).

Opción 2. Definir un cubo por dimensiones especificadas a lo largo de los ejes de coordenadas.
Argumentos: nombre del objeto y tres valores de las coordenadas Dx, Dy y Dz.

2. Cylinder (Cilindro)

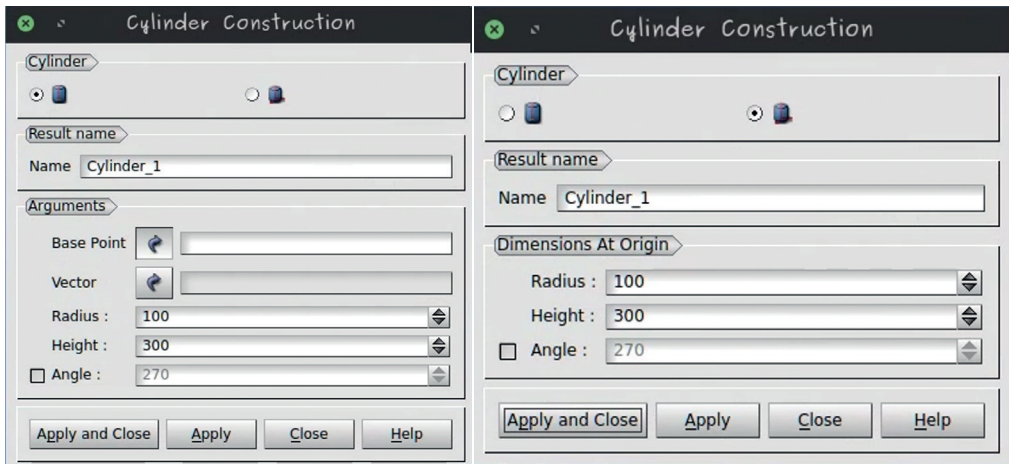


Figura 42. *Cylinder*

Opción 1. Definir un cilindro por un Punto Base (punto central de la base del cilindro), un vector (eje del cilindro) y sus dimensiones: el Radio y la Altura. De manera opcional se puede especificar el ángulo.
Argumentos: nombre del objeto, elegir vértice Base Point, elegir un vector, definir valores para las dimensiones: radio (Radius), altura (Height) y ángulo (Angle es opcional).

Opción 2. Definir un cilindro por el radio y la altura dados en el origen del sistema de coordenadas. Se puede especificar el ángulo para crear una porción de cilindro.
Argumentos: nombre del objeto, valores de las dimensiones: radio (Radius), altura (Height) y ángulo (Angle).

3. Sphere (Esfera)

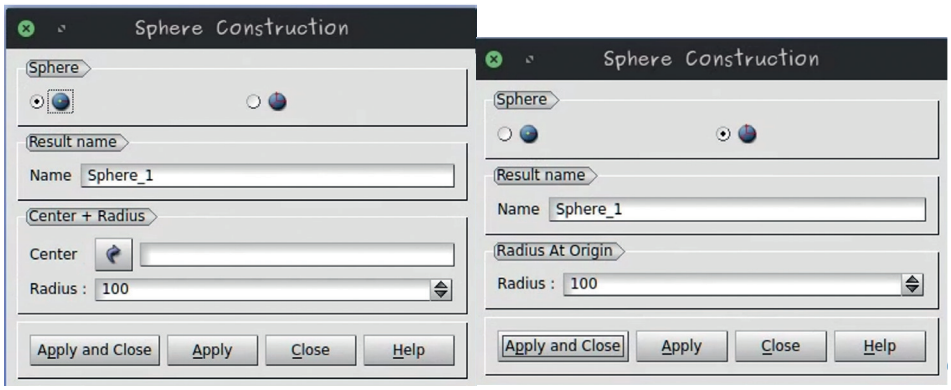


Figura 43. *Sphere*

Opción 1. Definir una esfera por el punto central y el radio.
Argumentos: nombre del objeto, vértice (para Center), valor al radio.

Opción 2. Definir una esfera con el centro en el origen del sistema de coordenadas por el radio.
Argumentos: nombre del objeto, valor del radio desde el origen.

4. Torus (Dona)

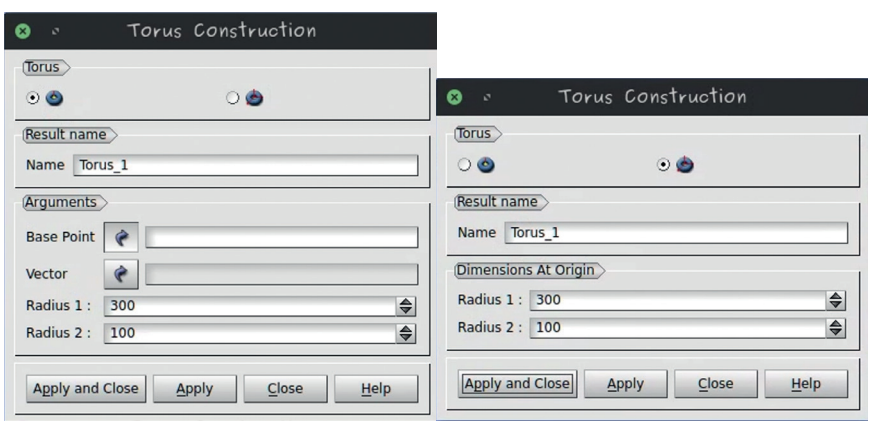


Figura 44. *Torus*

Opción 1. Definir un Torus por el Punto Base dado, el vector normal y los radios.
Argumentos: nombre del objeto, elige vértice (Base Point), elige un vector (para la dirección) y valores para los radios (Radius 1 and 2).

Opción 2. Definir un Torus con el centro en el origen de coordenadas por su radio.
Argumentos: nombre del objeto, valores de los radios (Radius 1 and 2).

5. Cone (Cono)

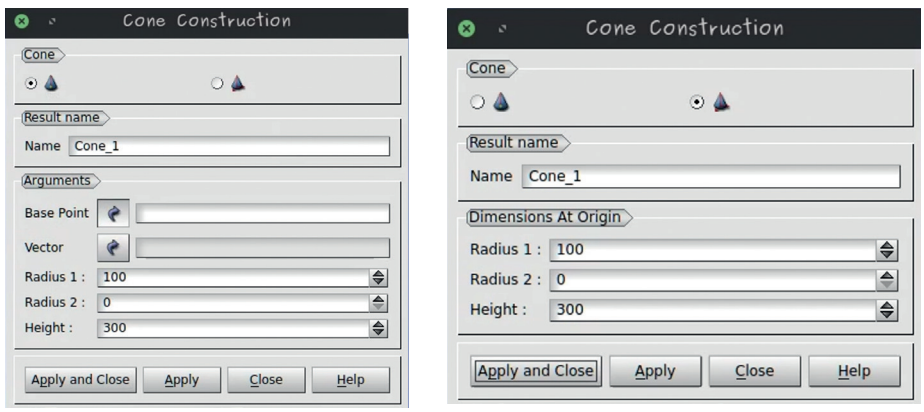


Figura 45. *Cone*

Opción 1. Definir un cono por el punto base (punto central de la base del cono), el eje, la altura y radio 1 y 2.
Argumentos: nombre del objeto, elige vértice (Base Point), elige un vector (para la dirección), indica los valores de los radios (Radius), establecer la altura (Height).

Opción 2. Definir un cono con el centro en el origen de las coordenadas por su altura y radios.
Argumentos: nombre del objeto, indica los valores del radio (Radius) y establecer la altura (Height).

6. Rectangle (Rectángulo)

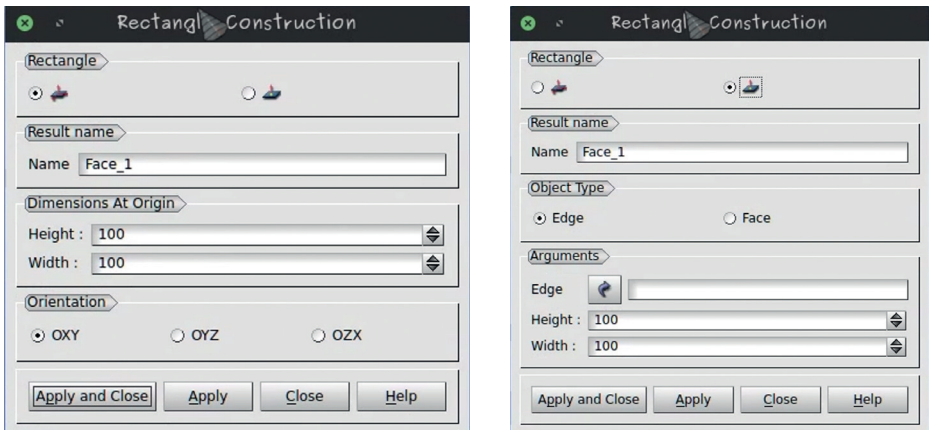


Figura 46. *Rectangle*

Opción 1. Crear un rectángulo en el origen de las coordenadas que definen sus límites por la altura y el ancho y su eje por los botones de radio de orientación (OXY, OYZ u OZX).
Argumentos: nombre del objeto, indica los valores de las dimensiones, altura (Height), ancho (Width) y orientación (OXY, OYZ u OZX).

Opción 2. Definir un rectángulo por la altura y tamaño, un borde que define el centro de la cara.
Argumentos: nombre del objeto, indica un vector (Edge), los valores de las dimensiones de altura (Height) y ancho (Width) para describir los tamaños de la cara.

7. Disk (Disco)

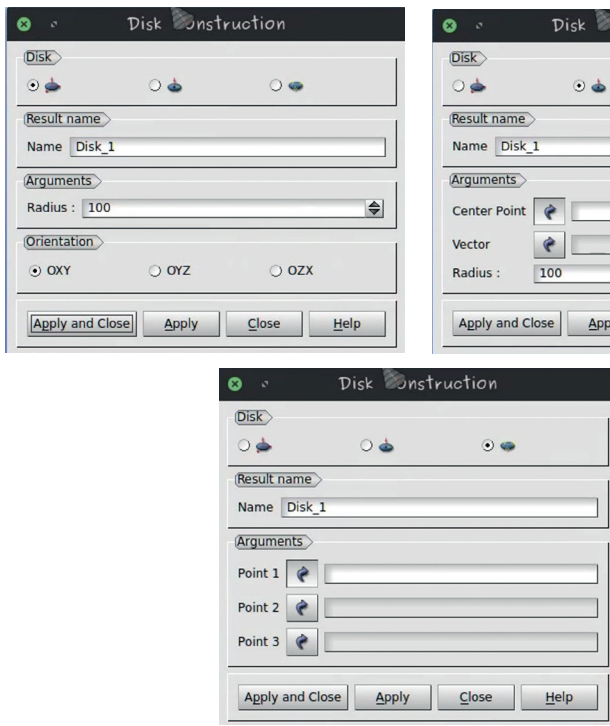


Figura 47. *Disk*

Opción 1. Crear un disco en el origen de las coordenadas que definen su radio y seleccionar la orientación de su eje (OXY, OYZ u OZX).
Argumentos: nombre del objeto, valor del radio y orientación (OXY, OYZ u OZX).

Opción 2. Definir un disco por un punto central, un vector que define la normal de un círculo y un radio.
Argumentos: nombre del objeto, elige vértice para el centro (Center Point), elige vector (para la dirección) y el radio.

Opción 3. Definir un disco por tres puntos que se encuentran en su límite.
Argumentos: nombre del objeto, definir los tres puntos que formarán el disco.

8. Pipe TShape (Forma de tubo)

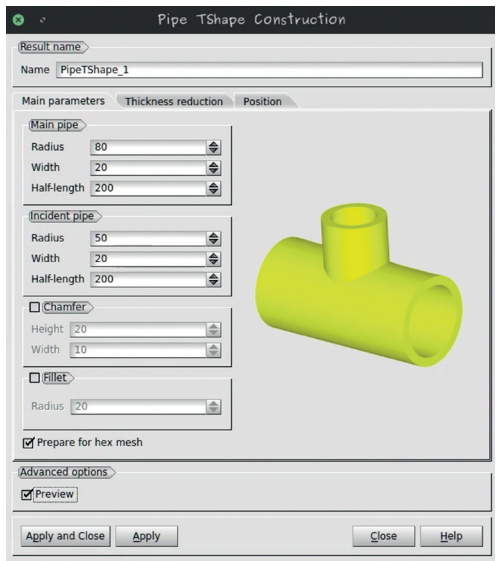


Figura 48. *Pipe TShape*

Parámetros principales:

- Tubo principal (Main Pipe). Permite definir el radio interno, el espesor y la longitud media.
- Tubo incidente (Incident Pipe). Permite definir el radio interno, el espesor y la longitud (la distancia entre el extremo del tubo incidente y el eje central del tubo principal).
- Chaflán. Permite crear un chaflán en la unión de las tuberías principal y de incidente. Sus parámetros son altura a lo largo del tubo incidente y anchura a lo largo del tubo principal.
- Filete. Permite crearlo en la unión de las tuberías principal y de incidente definiendo el radio del filete.
- HexMesh. Esta casilla permite dividir la forma en bloques adecuados para malla hexaédrica.

New Entity > Generation

Si nos dirigimos al menú **New Entity** > **Generation** nuevamente podremos observar seis opciones.

1. **Extrusion (extrusión)** es la propagación de la forma de base seleccionada en una cierta dirección y por una cierta distancia. Este ícono ayudará a ofrecer la forma del objeto. La herramienta cuenta con tres alternativas que se describen a continuación. Figura 49.

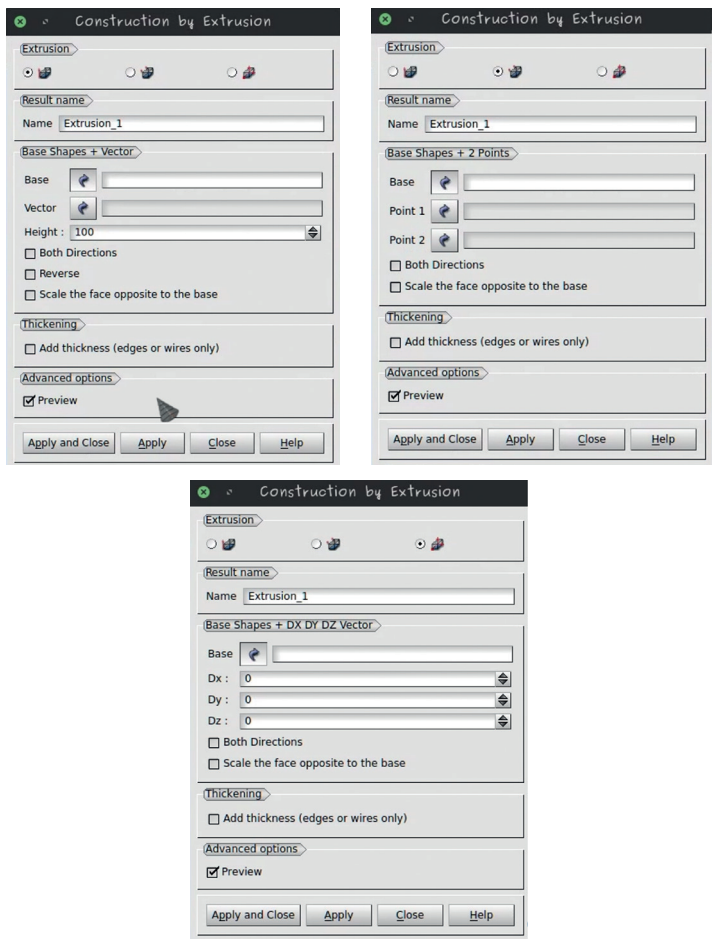


Figura 49. *Extrusion*

Opción 1. Definición de parámetros y opciones:

- Nombre de la forma resultante.
- Base. El objeto extruido.
- Vector que sirve para la dirección y la distancia.
- La casilla de ambas direcciones (Both Directions) permite extraer la forma de la base tanto hacia adelante como hacia atrás.
- La casilla de verificación inversa (Reverse) invierte la dirección del vector.
- La casilla de verificación (Scale), la cara opuesta a la base (Scale the face opposite to the base) y el factor de escala (Scale factor) permiten construir la extrusión con la base opuesta escalada.
- Agregue la casilla de espesor (Thickness) y el campo espesor permite agregar espesor al prisma creado.

Opción 2. Veamos los parámetros y opciones:

- Nombre de la forma resultante.
- Base. El objeto extruido.
- Punto 1 (Point 1). Indica la dirección y punto 2 (Point 2) indica la distancia.
- La casilla de ambas direcciones (Both Directions) permite extraer la forma de la base tanto hacia adelante como hacia atrás.
- La casilla de verificación (Scale), la cara opuesta a la base (Scale the face opposite to the base) y el factor de escala (Scale factor) permiten construir la extrusión con la base opuesta escalada.
- Agregue la casilla de espesor (Thickness) y el campo espesor permite agregar espesor al prisma creado.

Opción 3. Veamos los parámetros y opciones:

- Nombre de la forma resultante.
- Base. El objeto extruido.
- Ingresamos los valores necesarios con respecto a las direcciones Dx, Dy y Dz.

- La casilla de ambas direcciones (Both Directions) permite extraer la forma de la base tanto hacia adelante como hacia atrás.
- La casilla de verificación (Scale), la cara opuesta a la base (Scale the face opposite to the base) y el factor de escala (Scale factor) permiten construir la extrusión con la base opuesta escalada.
- Agregue la casilla de espesor (Thickness) y el campo espesor permite agregar espesor al prisma creado.

1. Revolution (revolución)

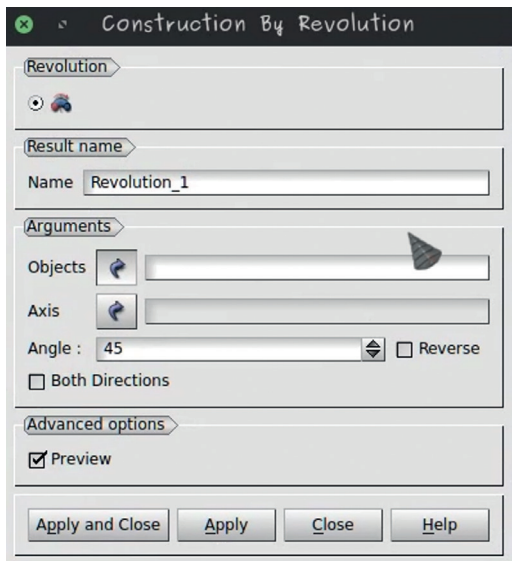


Figura 50. *Revolution*

Para crear una forma revolución, como se muestra en la figura 49, es necesario definir el objeto de origen a rotar (vértice, borde, alambre plano, cara o concha), el eje de revolución (Axis) que sirve para la dirección y el ángulo (Angle) por el cual la forma debe girarse alrededor del eje (en grados). Argumentos: La casilla Booth Directions permite extraer el objeto tanto hacia adelante como hacia atrás.

2. Filling (relleno)

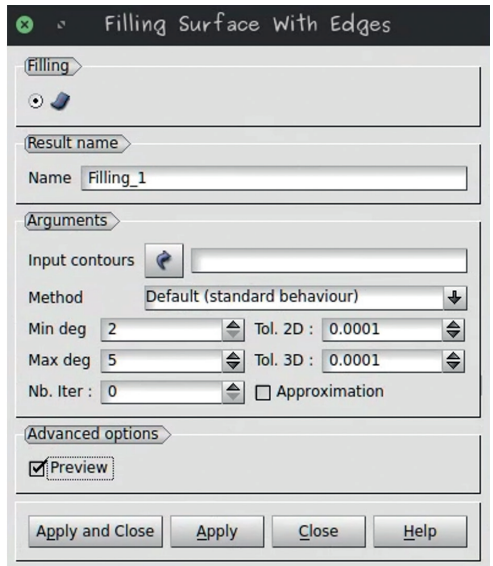


Figura 51. *Filling*

Para esta sección se deberán definir los siguientes parámetros:

1. Nombre del objeto resultante.
2. Contornos de entrada (Input Contours) lista de aristas / curvas que se utilizarán para crear la superficie.
3. Mínimo y máximo grado (Min deg y Max deg) de ecuación de las curvas BSpline o Besier que describen la superficie.
4. Tolerancia para 2D y para 3D (Tol. 2D y Tol 3D). Distancia mínima entre la superficie creada y los contornos de entrada.
5. Número de iteraciones (Nb. Iter) define el número máximo de iteraciones. Un mayor número de iteraciones permite producir una superficie mejor.
6. Existen tres tipos de métodos (Method) para realizar la operación de llenado:
   a) Por defecto: el comportamiento estándar.
   b) Utilizar la orientación de los bordes. Si se invierte un borde, la curva de este borde se invierte antes de ser utilizada.

c) La orientación de los bordes se corrige automáticamente. La orientación de las curvas se cambia para minimizar la suma de las distancias entre los extremos de los bordes.

7. Aproximación (Approximation). Las curvas BSpline se generan en el proceso de construcción de la superficie. El uso de la aproximación ralentiza el sistema, pero permite construir la superficie para casos complejos.

3. Extrusion Along Path (Extrusión de un objeto a lo largo de una ruta)

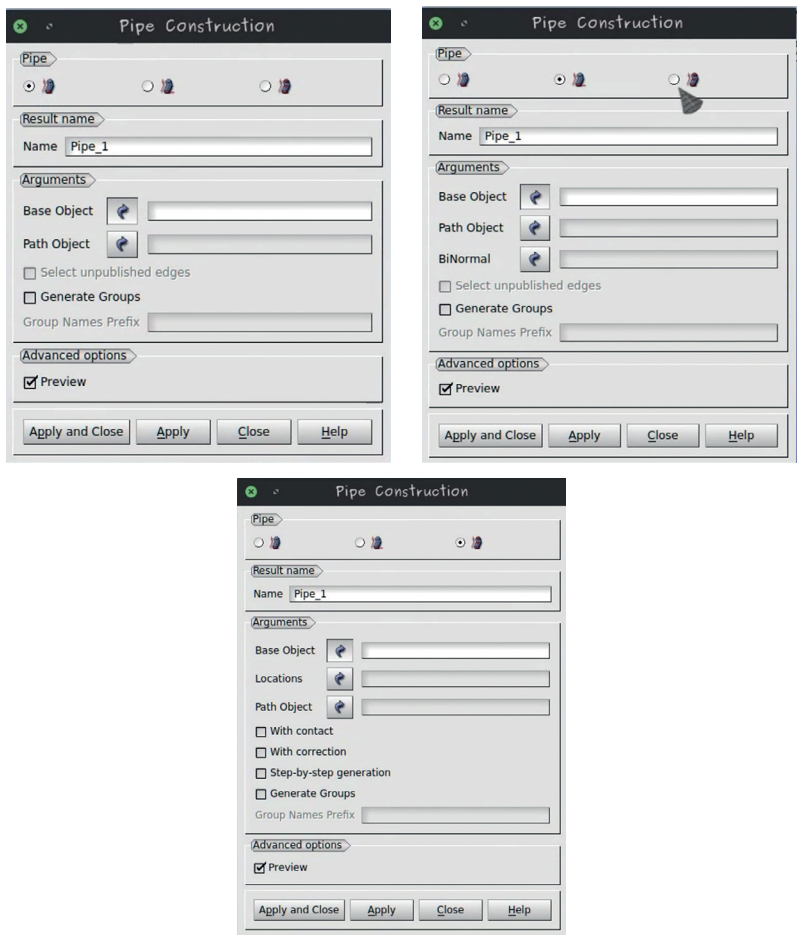


Figura 52. *Extrusion Along Path*

Opción 1. Definir el objeto base (vértice, borde, cable plano, cara o shell) que se extruirá y el objeto de trayectoria (borde) a lo largo del cual se extruirá el objeto base.
Argumentos: nombre del objeto, selección de una forma (vértice, borde, alambre plano, cara o concha) que sirve como objeto base, elección de una forma (borde o cuerda) para la definición de la ruta.

Opción 2. Definir el objeto base (borde, cuerda, plano o cara), que será extruido, el objeto que servirá de ruta (borde o cuerda) a lo largo del cual se extruirá el objeto base y el vector BiNormal (borde, cuerda o vector).
Argumentos: nombre del objeto, elegir una forma (borde, cuerda plano o cara) que sirve como objeto base, elegimos una forma (borde o cuerda) para definir la trayectoria (Path object), elegir una forma (borde o cuerda) para establecer una dirección BiNormal fija para realizar la extrusión.

Opción 3. Es posible seleccionar adicionalmente los perfiles del objeto a construir. Para ello, puede definir los siguientes parámetros:

Nombre del objeto resultante:

Objeto base o lista de objetos (aristas, cuerdas, caras o conchas) que serán extruidos.

Ubicaciones (Locations): un vértice o una lista de vértices que especifican las ubicaciones de objetos de base extruidos en el objeto de trayectoria resultante.

Path Object (borde o alambre), a lo largo del cual se extruirá el objeto base.

Controles adicionales:

Con el contacto (With contact), la sección se traduce en contacto con la columna vertebral.

Con corrección (With correction), la sección se gira para ser ortogonal (que forma ángulo recto) a la tangente de la espina en el punto correspondiente.

Comprueba la generación paso a paso (Step-by-step generation), el resultado se crea paso a paso, es decir, crea tubos entre cada par de secciones y los fusiona en una sola.

4. Restore Path (Restaurar la ruta)

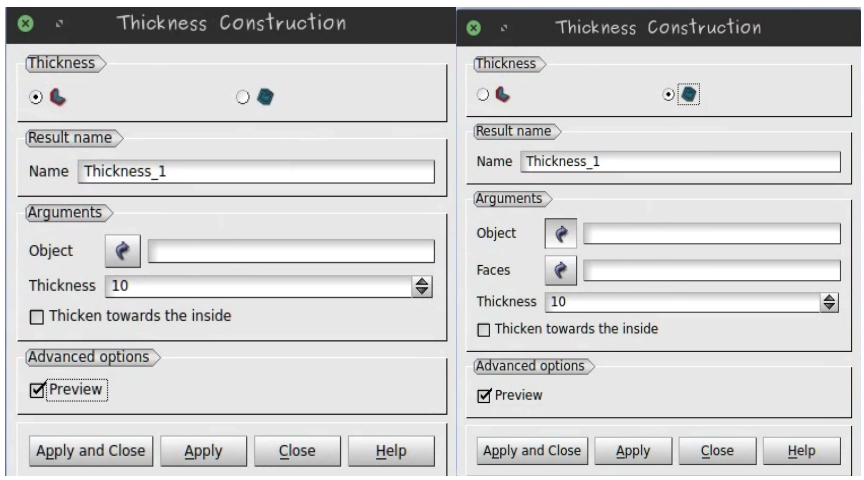


Figura 53. *Restore Path*

Para obtener el camino de una forma similar a una tubería, se debe establecer la cáscara de tipo de tubo o sólido y dos bases de tubo, que pueden definirse por una cuerda, una cara o una lista de aristas.
Argumentos: en seguida se debe seleccionar la casilla de verificación de bordes no publicados (Selected unpublished edges), que permitirá elegir en el visor los bordes que no se publican en el Explorador de objetos.
Argumentos: nombre del objeto, elegir una forma de tubo, cáscara o sólido (Pipe-like shell or solid), se debe elegir una forma de borde, cuerda o cara (First base face/wire/edge) para la primera base, y elegir una forma borde, cuerda o cara (Last base face/wire/edge) para la última base.

5. Thickness (Grosor)

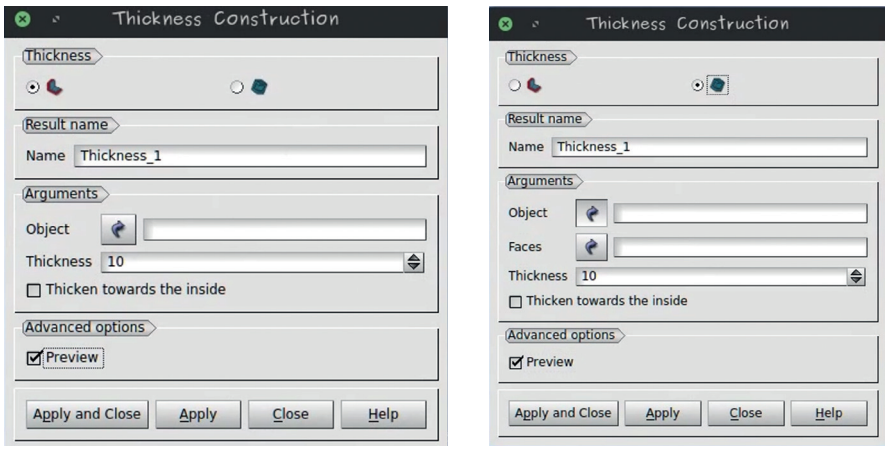


Figura 54. *Thickness*

Opción 1. El espesor se puede aplicar a una cara o a un borde (Shell) para crear un sólido.

Es necesario definir un objeto (Face o Shell) y el valor del espesor (Thickness). Espesar hacia el interior de la casilla (Thicken towars the inside) para permitir cambiar la dirección del grosor.

Opción 2. El espesor se puede aplicar a un sólido para crear un sólido hueco.

Es necesario definir un objeto sólido (Object), generar las caras (Face) que se quitarán e indicar el grosor (Thickness). Seleccionar la casilla Espesar hacia el interior (Thickness) que permite cambiar la dirección del grosor si es necesario.

New Entity > Explode

Al abrir el menú **New Entity** > **Explode**, se desplegarán las siguientes opciones:

1. **Edge** sirve para extraer los depósitos o mostrar simplemente los bordes. Figura 55.

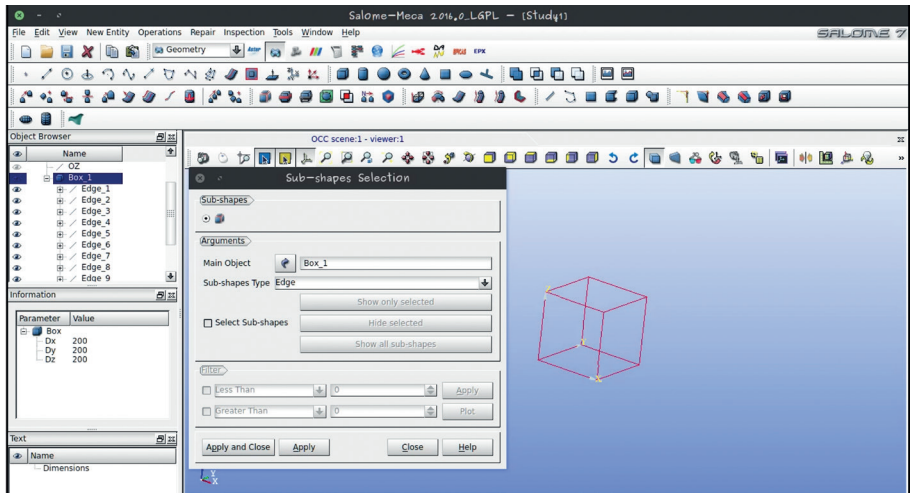


Figura 55. *Edge*

Para crear una lista de sub-formas (Edge), se debe definir el objeto principal (Main object) que se explotará, elegir el tipo de Sub-formas que se desea obtener; en este caso será Edge; aplicamos y cerramos.

2. **Face** es útil para extraer las caras. Figura 56.

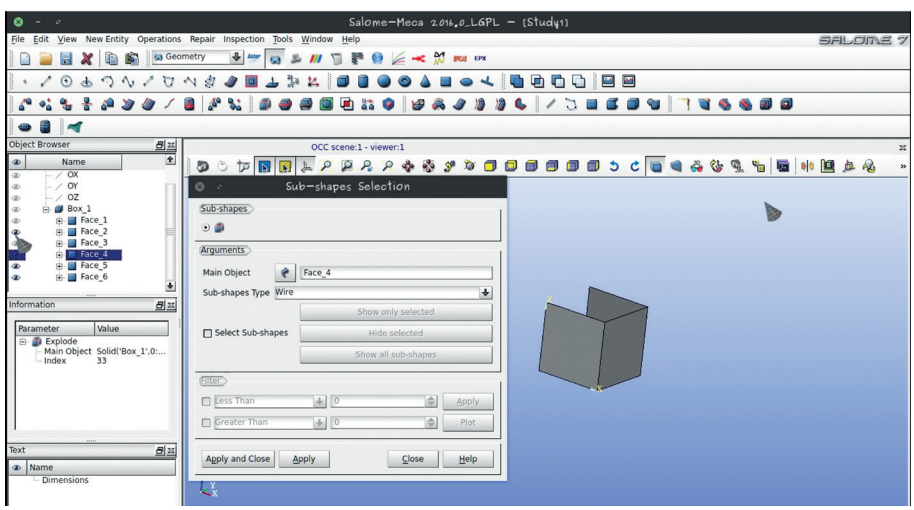


Figura 56. *Face*

Para crear una lista de sub-formas (Face), se debe definir el objeto principal (Main object) que se explotará, elegir el tipo de Sub-formas; en este caso será Face, aplicamos y cerramos.

3. **Shell** sirve para extraer sólidos. Figura 57.

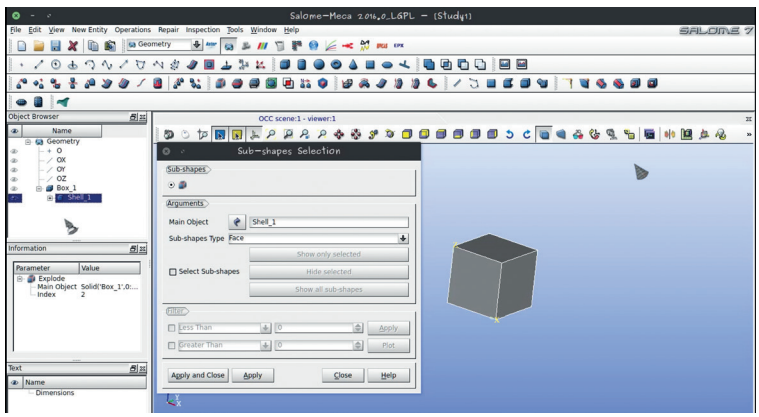


Figura 57. *Shell*

Para crear una lista de sub-formas (Shell), se debe definir el objeto principal que se explotará (Main object), elegir el tipo de Sub-formas que desea obtener de ese menú, en este caso será Shell, aplicamos y cerramos.

4. **Vertex** es útil para extraer los vértices. Figura 58.

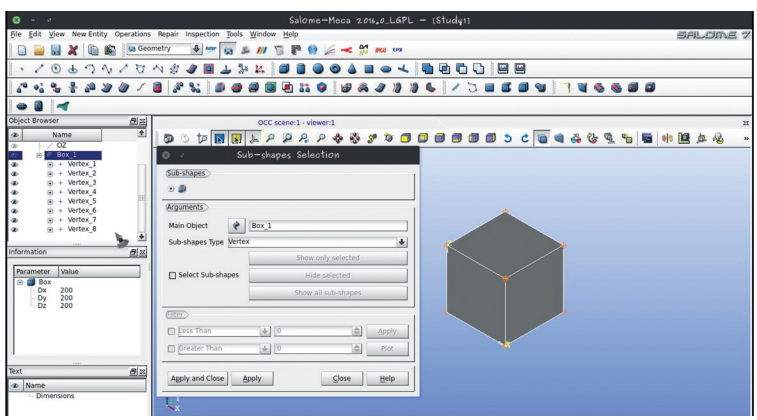


Figura 58. *Vertex*

Para crear una lista de sub-formas (Vertex), es conveniente considerar el objeto principal que se explotará (Main object); elegir el tipo de sub-formas que se desea obtener, en este caso será Vertex, aplicamos y cerramos.

5. **Wire** funciona para extraer una figura plana, pero únicamente los bordes. Figura 59.

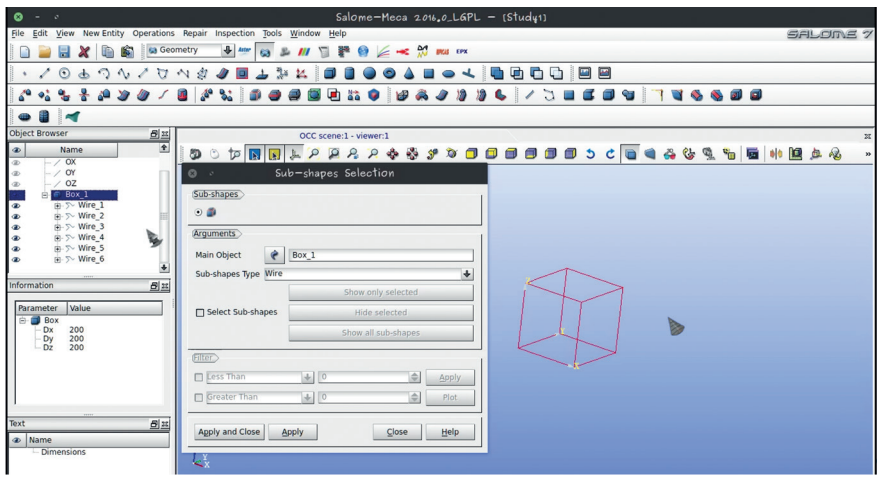


Figura 59. *Wire*

Para crear una figura plana, se requiere un paso semejante al anterior, se acude a la lista de sub-formas (vertex), se debe definir el objeto principal (Main object), elegir el tipo de sub-formas, se aplica y se cierra.

New Entity > Build

Ahora el programa presentará un nuevo menú: New Entity > Build, mediante el cual podremos observar algunas opciones.

1. Edge (Crear borde)

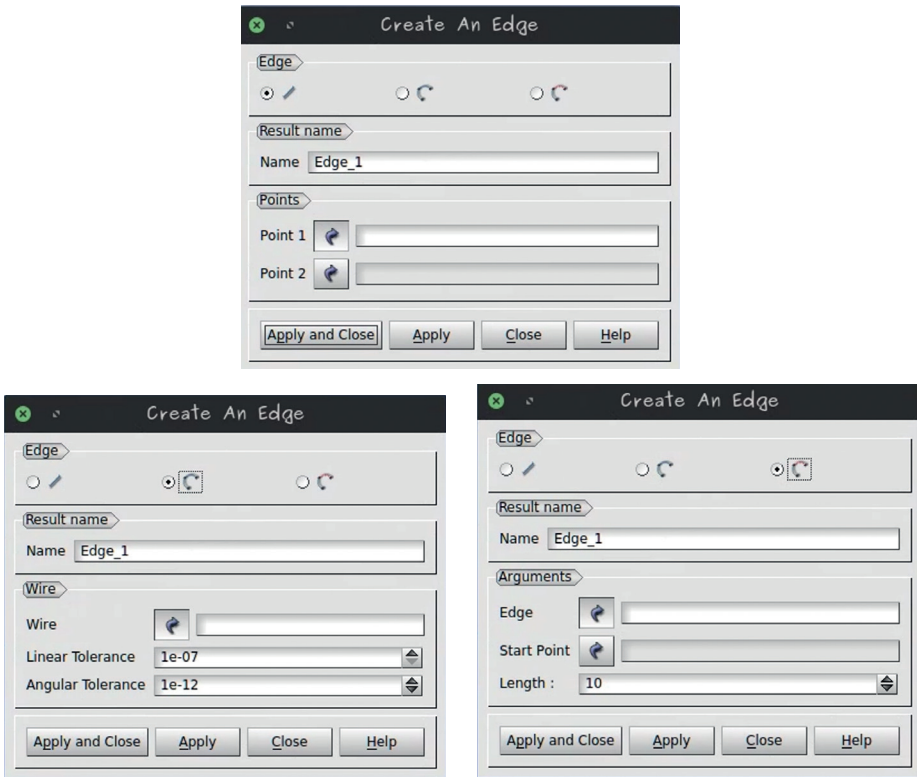


Figura 60. *Edge*

Opción 1. Crear un borde lineal especificado por dos puntos (Point 1 y Point 2), que son el primero y el último vértice del borde.
Argumentos: nombre del objeto, elegir dos vértices (Point 1 y 2).

Opción 2. Crear un borde especificando un solo hilo.

En este modo son posibles los siguientes casos de uso:

- Todos los bordes que forman el hilo se encuentran en la misma curva geométrica, es decir curva (borde1) == curva (borde2).
- Los bordes que forman el hilo se encuentran en curvas analíticas del mismo tipo, por ejemplo segmentos de línea, arcos, etcétera.
- Los bordes que forman el hilo tienen la misma tangencia en los puntos de conexión.

Argumentos: nombre del objeto, elige un hilo (Wire), indicar la tolerancia lineal (Linear Tolerance), indicar la tolerancia angular (Angular Tolerance).

Opción 3. Construir un borde de la longitud requerida en cualquier borde existente.

El parámetro Start Point es opcional:

- Si se utiliza, permite seleccionar cualquier punto existente.
- Si se omite, se utiliza el punto inicial del borde.

La longitud del borde (Length) puede ser negativa.

Argumentos: nombre del objeto, elige un borde e indica la longitud y elige un vértice.

2. Wire (Crear un hilo)

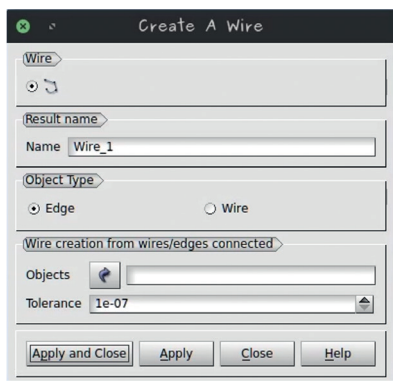


Figura 61. *Wire*

Con Wire puede crear un cable desde varios bordes o cables conectados seleccionándolos en el navegador de objetos o en el área de trabajo:

- Se inicia con el nombre de nuestro objeto resultante.
- Es posible seleccionar hilos (Wire) o bordes (Edge) para especificar el tipo de objeto.
- Elija los objetos conectados para crear nuestro hilo.
- Indique una tolerancia (Tolerance) que se utilizará para comprobar las conexiones.

3. Face (Crear una cara)

Este ícono cuenta con tres opciones que a continuación se describen:

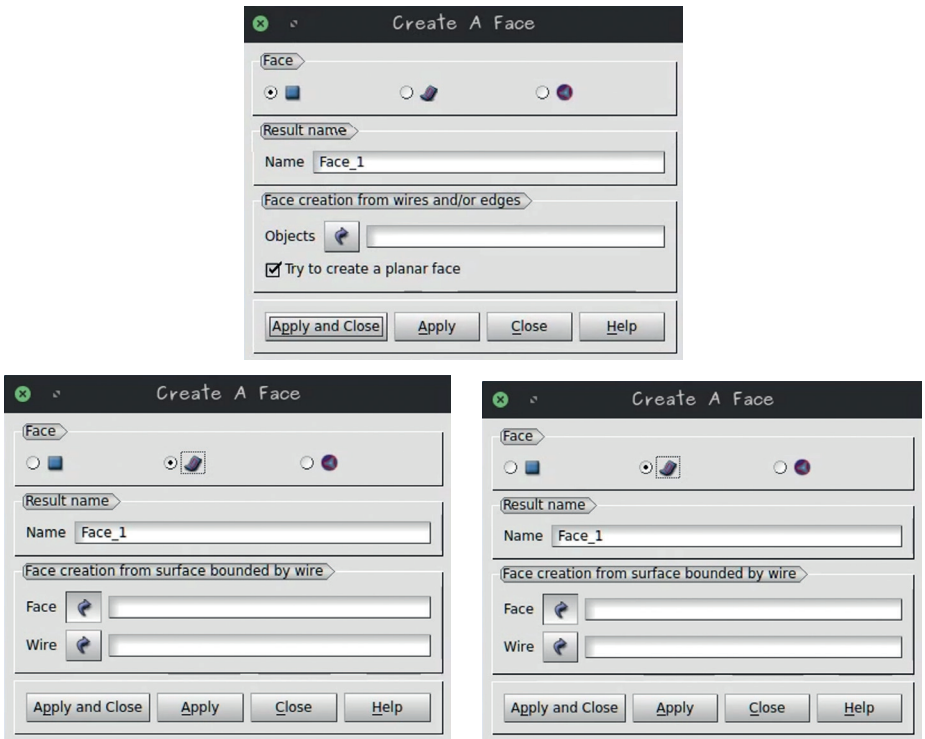


Figura 62. *Face*

Opción 1. Crea una cara (Face) para lo cual es necesario seleccionar la(s) forma(s) de entrada.
Argumentos: nombre del objeto y elegir un hilo o forma simple.

Opción 2. Crea una cara basada en la superficie de otra cara y limitada por un hilo.
Argumentos: nombre del objeto, elige una cara (puede ser un plano), elige un hilo (Wire).

Opción 3. Crea una cara especificando un conjunto de aristas formando un hilo cerrado y restricciones:

- Especifique un cable de entrada seleccionándolo en el navegador de objetos o en el área de trabajo.
- Especifique las restricciones asociando las caras con los bordes.

Argumentos: nombre del objeto, elegir lista de bordes de entrada y caras de restricción. Si falta una cara de restricción para algún borde, significa que no hay ninguna restricción asociada a este borde.

Nota: conjunto de bordes debe formar un alambre cerrado.

4. Shell (crear un Shell a partir de un compuesto de caras o una lista de caras)

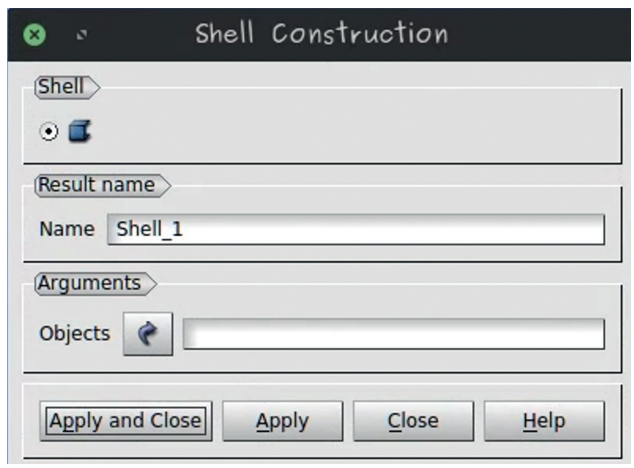


Figura 63. *Shell*

Argumentos: nombre del objeto, elige compuesto de caras o lista de caras que tienen bordes conectados (Objects).

5. Solid (sólido)

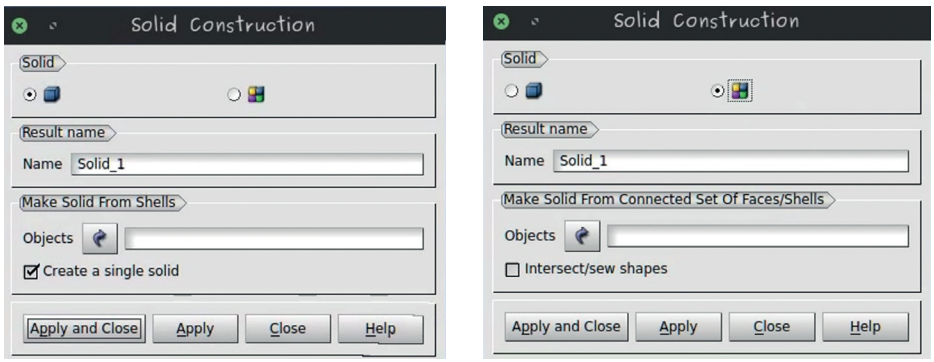


Figura 64. *Solid*

Opción 1. Crea un Solid desde una lista de cáscara (Shell).
Argumentos: nombre del objeto, elige una cáscara cerrada o una lista de cáscaras cerradas (Shell).

Opción 2. Crea un Solid (o un compuesto de sólidos) a partir de una lista de caras o cáscaras conectadas.
Argumentos: nombre del objeto, selecciona un conjunto de caras conectadas o cáscaras.

6. Compound (Compuesto)

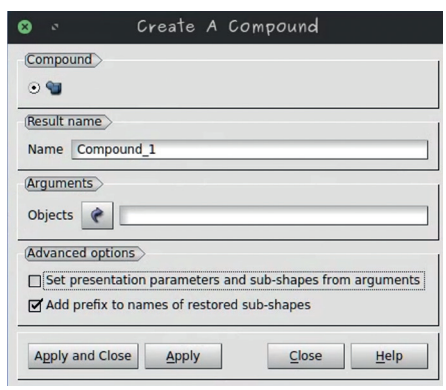


Figura 65. *Compound*

Puede crear un compuesto a partir de una lista de formas.

Argumentos: nombre del objeto, selecciona lista de formas (Objects).

Puede hacer uso de las opciones avanzadas si son necesarias.

Ahora veremos algunas herramientas del menú Operations, y describiremos brevemente el funcionamiento de algunas de ellas ilustrando los posibles resultados del uso de dichas herramientas; de la misma forma pueden ser utilizadas en elementos 2D y 3D.

Menú Operations

Operations > Boolean

Nos dirigimos a **Operations** > **Boolean** para encontrar las siguientes cuatro funciones:

1. **Fuse (Fusible)** es una operación que crea una forma a partir de una lista de objetos. Figura 66.

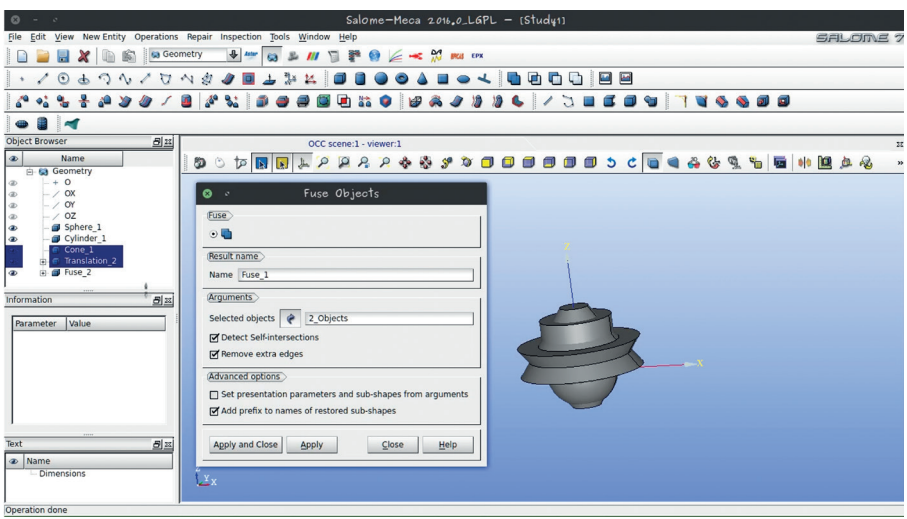


Figura 66. *Fuse*

En el cuadro de diálogo deberá ejecutar lo siguiente:

- Ingresar o aceptar el nombre de la forma resultante.
- Haga clic en la flecha y seleccione los objetos a unir.
- Active la casilla si desea Detectar Auto-intersecciones.
- Active la casilla si desea eliminar los bordes.
- Puede hacer uso de las opciones avanzadas.
- Pulse el botón Aplicar, o Aplicar y cerrar para obtener el resultado.

2. **Common** (Común) la operación corta la parte común de una lista de objetos y la transforma en un objeto nuevo. Figura 67.

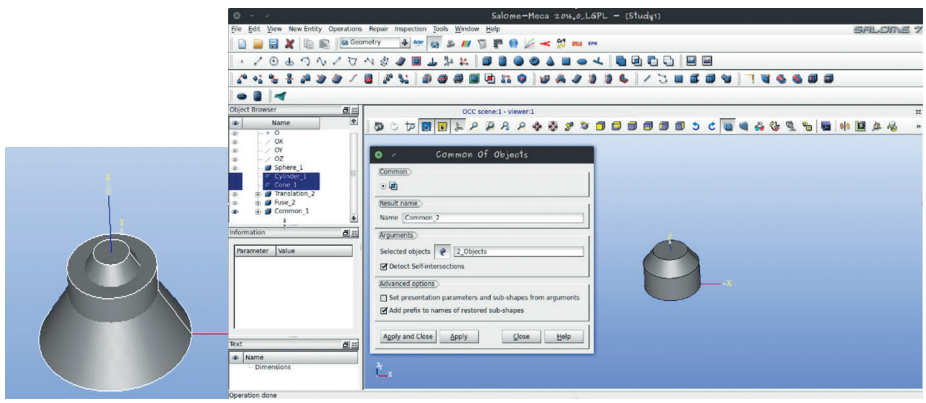


Figura 67. *Common*

En este cuadro de diálogo:

- Ingrese o acepte el nombre de la forma resultante.
- Haga clic en el botón de flecha y seleccione los objetos.
- Active la casilla si desea Detectar Auto-intersecciones.
- Pulse el botón Aplicar, o Aplicar y cerrar para obtener el resultado.

3. **Cut** es la operación que corta un objeto en las formas deseables. Figura 67.

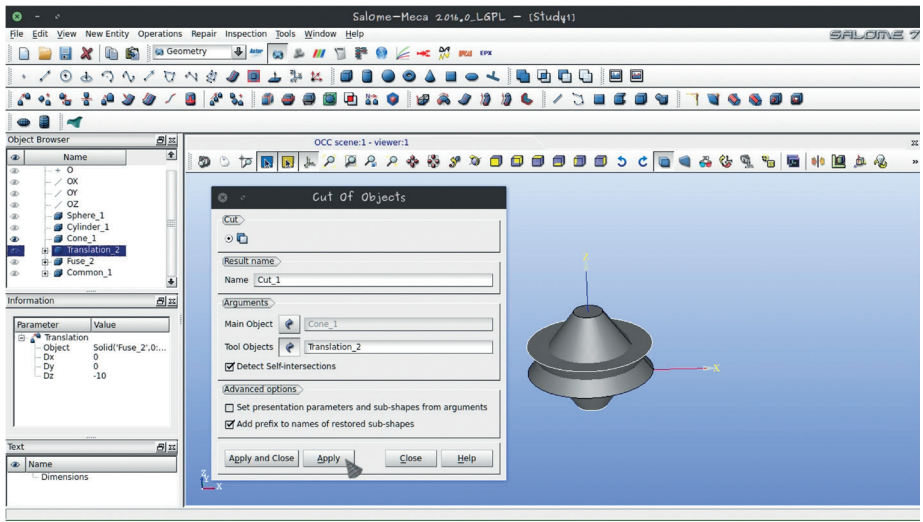


Figura 68. *Cut 1*

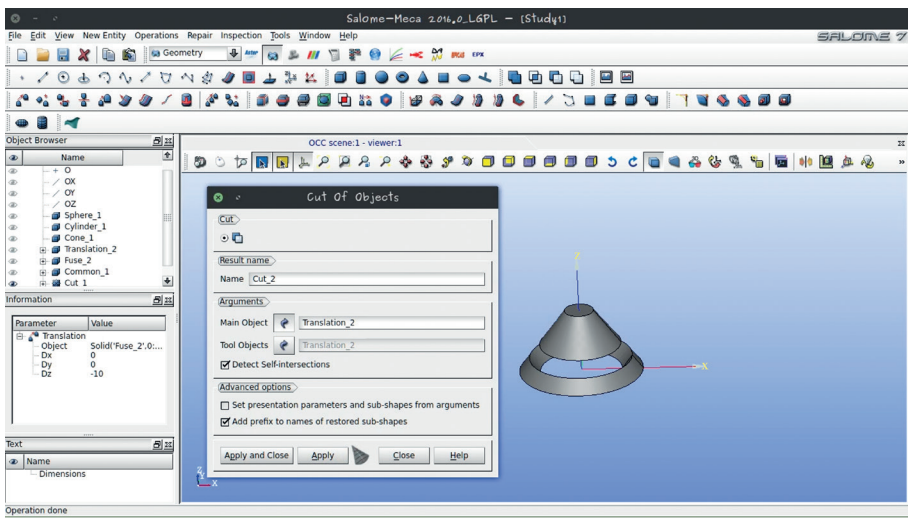


Figura 69. *Cut 2*

En el cuadro de diálogo, se deberá proceder con las siguientes ejecuciones:

- Ingrese o acepte el nombre de la forma resultante.
- Haga clic en el botón de flecha de Main Object y seleccione el objeto principal, que será cortado por el objeto de Tool Objects.

- Active la casilla si desea Detectar Auto-intersecciones.
- Use las funciones avanzadas si desea.
- Pulse el botón Aplicar, o aplique y cierre para obtener el resultado.

4. **Intersection (Intersección)** es una operación que crea un borde compuesto de ellos y justamente representa la intersección de dos formas.

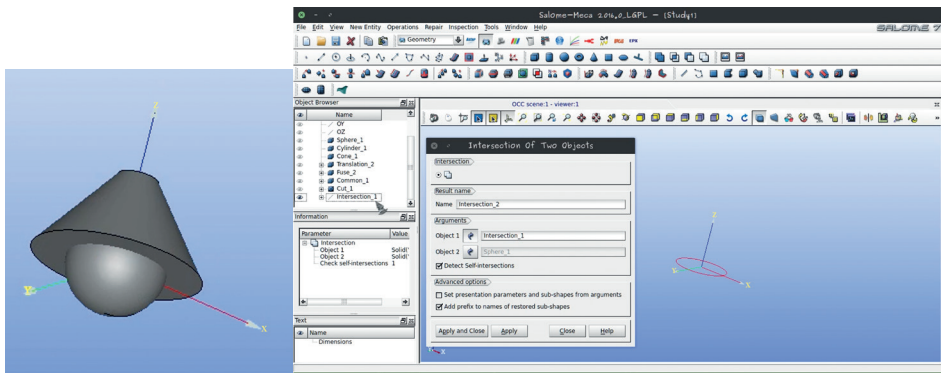


Figura 70. *Intersection*

En el cuadro de diálogo:

- Ingrese o acepte el nombre de la forma resultante.
- Haga clic en el botón de flecha y seleccione los objetos que se intersectan.
- Active la casilla si desea Detectar Auto-intersecciones.
- Use las funciones avanzadas si desea.
- Pulse el botón Aplicar, o aplique y cierre para obtener el resultado.

Operations > Transformation

Una vez concluido lo anterior, nos dirigimos a Operations > Transformation en donde encontraremos las siguientes seis operaciones, algunas de ellas con varias opciones a elegir según los requerimientos del usuario.

1. **Translation (Translación)** esta operación hace un traslado de un objeto.

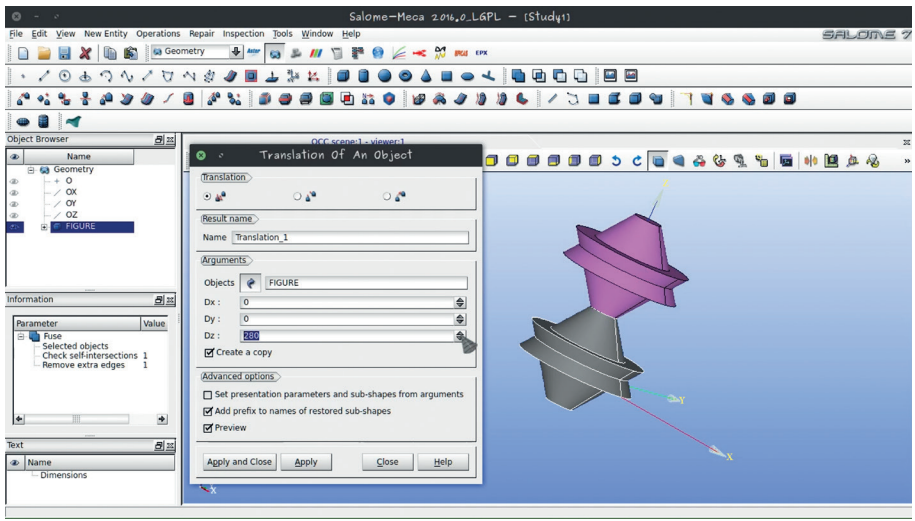


Figura 71. *Translation*

Esta operación tiene 3 opciones de uso:

Opción 1. Puede definir por coordenadas a lo largo de los ejes, DX, DY, DZ y sobre los Objects, elegimos el objeto a trasladar. Podemos modificar las opciones avanzadas, y aplicamos, o aplicamos y cerramos para obtener el resultado.

Opción 2. Es posible definir el lugar por dos puntos (Point 1 y Point 2).

Opción 3. Se puede definir un vector.

2. **Rotation (Rotación).** Como su nombre lo indica esta operación gira el objeto y cuenta con dos vías.

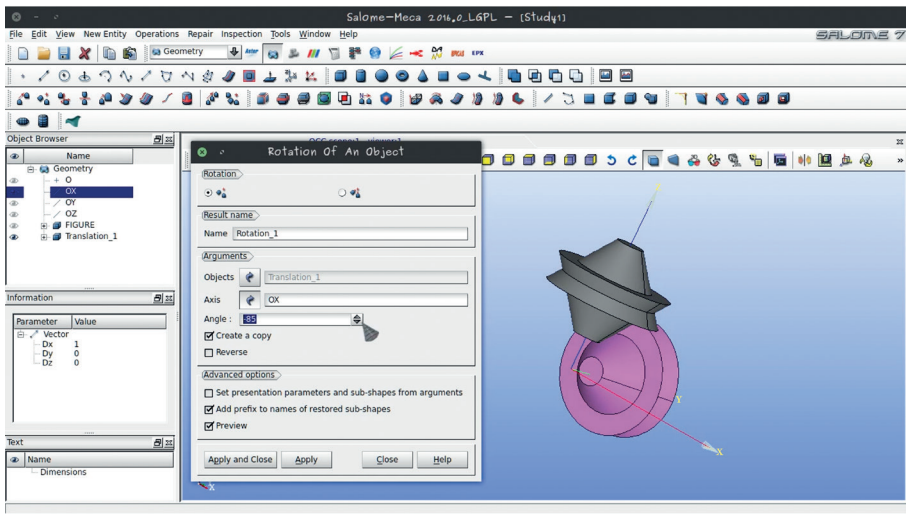


Figura 72. *Rotation*

Opción 1. Aceptar nombre o cambiarlo, definir el objeto a rotar, elegir el eje de rotación y un ángulo de rotación.

La casilla de verificación Reverse permite ir en dirección contraria.

La casilla Create a copy permite mantener el objeto inicial, de lo contrario se eliminará.

Opción 2. Permite definir el Objeto girado por tres puntos. El eje de rotación será a través del Punto Central, el Ángulo de rotación es desde el Punto Central al Punto 1 y el Punto 2.

3. **Mirror Image** es una operación diseñada para creae una copia simétrica del objeto, que se puede reflejar de tres maneras. Figura 73.

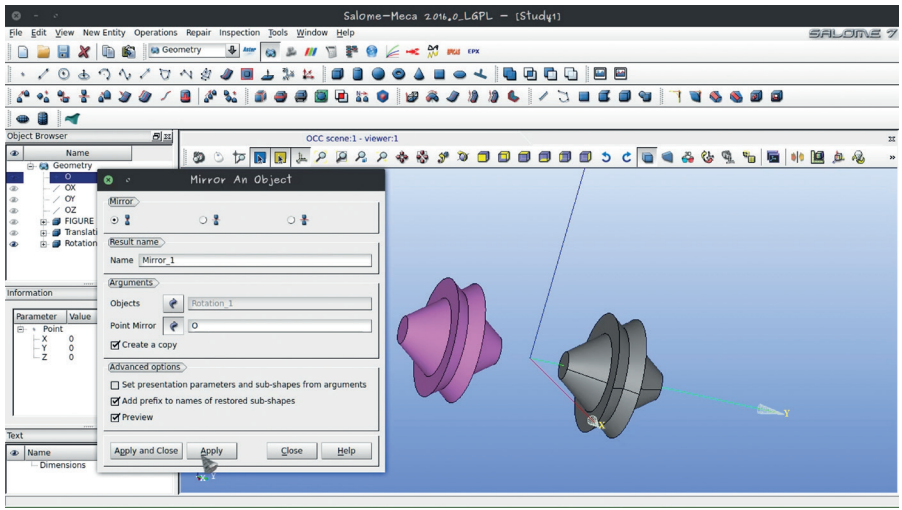


Figura 73. *Copia simétrica del objeto*

Opción 1. El objeto puede ser reflejado a través de un punto de simetría
Argumentos: nombre, elegir uno o varios objetos, vértice o eje.

Opción 2. Puede ser reflejado a través de un eje.
Argumentos: nombre, elegir uno o varios objetos, eje.

Opción 3. Puede ser reflejado a través de un plano.
Argumentos: nombre, elige uno o varios objetos, elige el plano.

Al finalizar podemos aplicar, o aplicar y cerrar para obtener el resultado.

4. **Scale Transform** es una operación que crea una forma escalada del objeto elegido. Se puede realizar de dos maneras como se muestra en la figura 74.

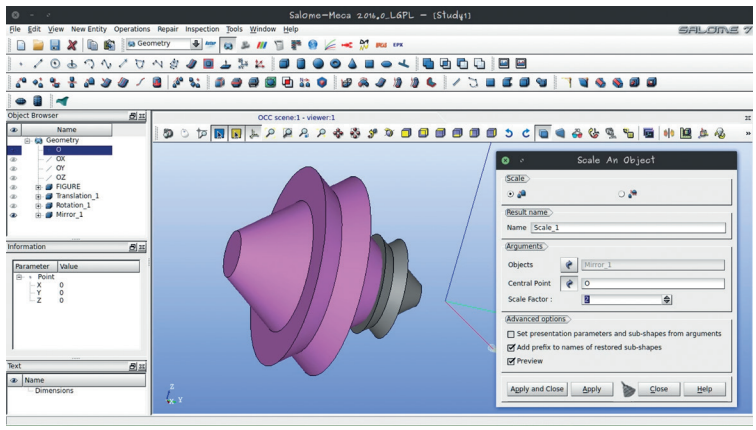


Figura 74. *Forma escalada del objeto*

Opción 1. En la escala, simple sus dimensiones cambian uniformemente.

Definamos los siguientes parámetros y opciones:

- Nombre del resultado.
- Objetos a ser escalados.
- Punto central (opcional) si este no es definido, la escala se realizará en el origen del sistema de coordenadas global.
- Factor de escala valor a ser escalado.
- Usar opciones avanzadas si son necesarias.

Opción 2. La escala múltiple permite escalar por diferentes factores a lo largo de los ejes:

- Nombre del resultado.
- Objetos a ser escalados.
- Punto central (opcional) en relación con el objeto escalado.
- Factor de escala sobre eje X / Y / Z, dimensión a escalar.
- Usar opciones avanzadas si son necesarias.

Al término de nuestros parámetros, podremos aplicar, o aplicar y cerrar.

5. **Multi-Translation** crea múltiples copias de un objeto en dos direcciones. Figura 75.

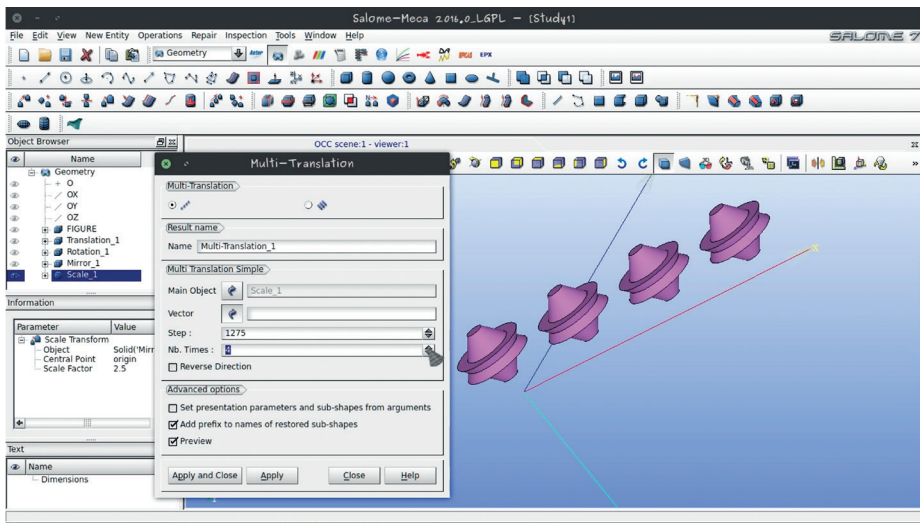


Figura 75. *Una dirección*

Opción 1. Una dirección se puede definir los siguientes parámetros y opciones. Figura 76:

- Nombre del resultado.
- Objeto a realizar con dicha operación.
- Se define un vector.
- Se marca el Step (Paso) para dar la distancia entre las copias.
- Se indica Nb. Time que se refiere al número de copias.
- La casilla dirección inversa permite cambiar la dirección.
- Se pueden usar opciones avanzadas si son necesarias.

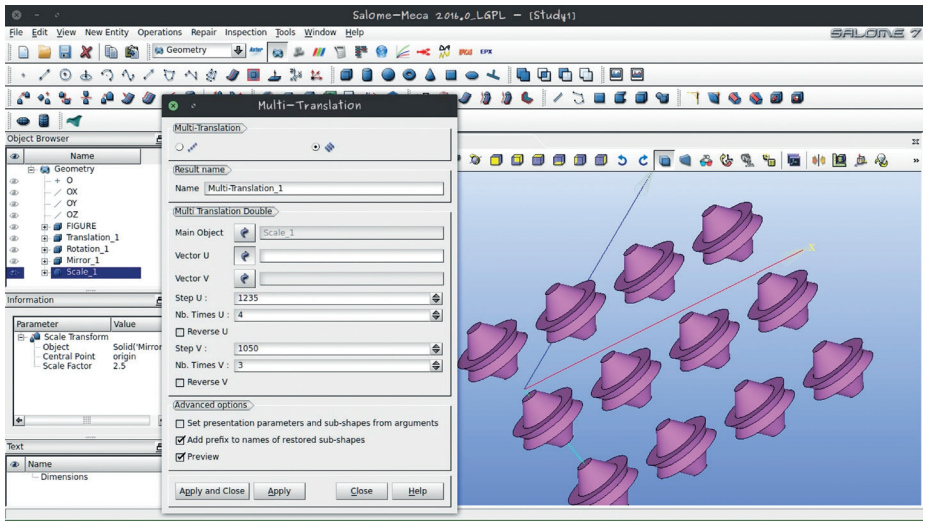


Figura 76. *Doble dirección*

Opción 2. La doble dirección, como se visualiza en la figura 77, permite generar los siguientes parámetros y opciones:

- Nombre del resultado.
- Objeto para dicha operación.
- Se eligen los vectores U y V (DX y DY, por defecto).
- Se elige la distancia entre las copias en las diferentes direcciones, Step U y V.
- Se indica Nb. Time U y V se refiere al número de copias del objeto.
- Reverse U y V permite cambiar la dirección de la traducción.

Una vez completados los parámetros y opciones anteriores, aplicamos, o aplicamos y cerramos.

6. **Multi-Rotation (Rotación múltiple)** crea un conjunto de objetos girados de un objeto, a partir de dos opciones.

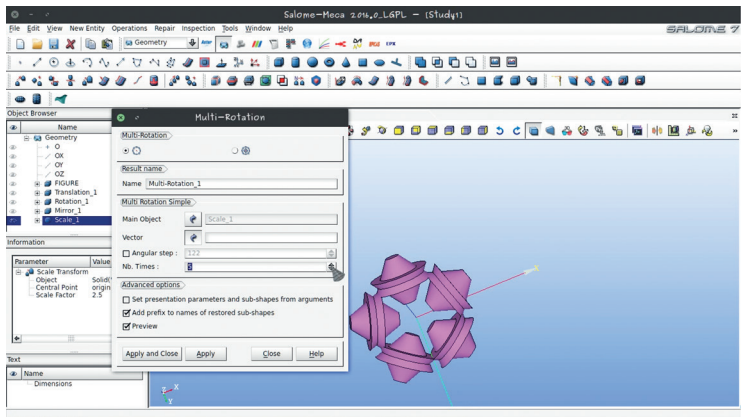


Figura 77. *Multi-Rotation*

Opción 1. Con la múltiple rotación simple definimos los parámetros y opciones. Figura 78:

- Nombre del resultado.
- Objeto a girar.
- Vector que define el eje de rotación (DZ por defecto).
- Angular Step (paso), es el ángulo por el cual se gira el objeto.
- Nb. Times es el número de copias de forma.
- Uso de opciones avanzadas, si son necesarias.

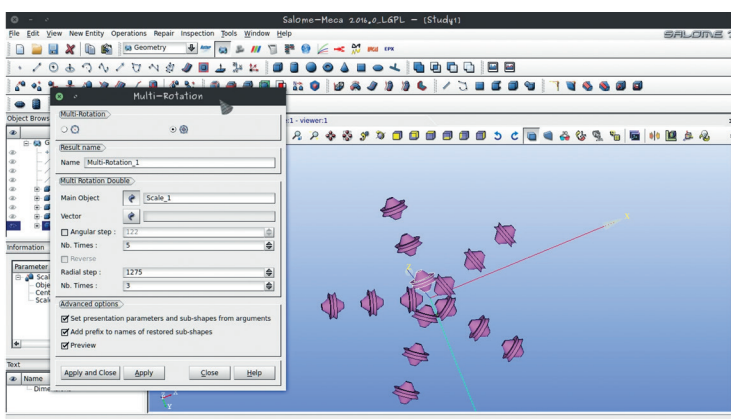


Figura 78. *Múltiple rotación doble*

Opción 2. Con la múltiple rotación doble definiremos los parámetros y opciones. Figura 78:

- Nombre del resultado.
- Objeto a girar.
- El vector define el eje de rotación (DZ por defecto).
- Angular Step es el ángulo por el cual se gira el objeto.
- Nb. Times 1 es el número de copias de forma rotada.
- La casilla inversa permite cambiar la dirección.
- Radial Step es la distancia entre las copias en la misma dirección.
- Nb. Times 2 es el número de copias de forma en la misma dirección.

Al término de los parámetros aplicamos, o aplicamos y cerramos. Enseguida viene un ícono muy importante con una cantidad amplia de funciones que explicamos ahora.

Extruded Cut (Extruir) 

Con esta herramienta podremos acceder en **New Entity > Extruded Cut.** La operación Extruir corte permite quitar fácilmente el material de un sólido mediante la extrusión de un perfil. Figura 79.

Para producir el corte extruido, se requiere:

- Un boceto dibujado en una cara plana del objeto que desea cortar.
- Cualquier borde cerrado o alambre de la forma deseada (círculo, elipse...).

Los argumentos son:

- Initial Shape. El objeto que desea cortar debe ser un sólido o un compuesto hecho de un solo sólido.
- Profile. El perfil debe ser un borde cerrado o hilo y debe ser plano.
- Height. Se refiere a conseguir la altura de la extrusión.
- Draft angle. Ángulo de inclinación en grados, se activa haciendo clic en este ícono: 
- Change direction. Para invertir dirección se presiona el botón: 

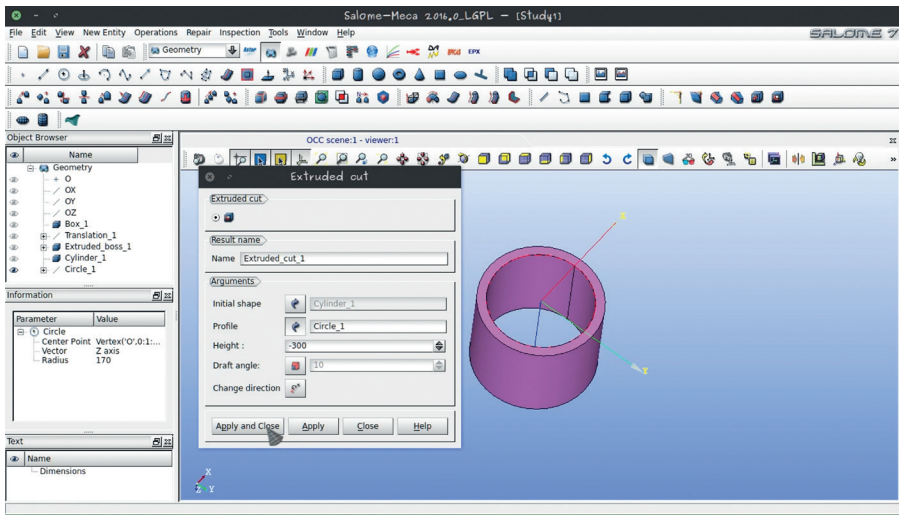


Figura 79. *Extruded Cut*

Extruded boss (Jefe extruido)

Posteriormente, accedemos al menú **Operations > Extruded boss**. La operación Extruir saliente le permite añadir fácilmente el material en un sólido. Figura 80.

Para producir el jefe extruido, se requiere:

- Un boceto dibujado en una cara plana.
- Cualquier borde cerrado o alambre de esta forma (círculo, elipse...).

Los argumentos son:

- Initial Shape. Objeto al que desea agregar material. Debe ser un sólido o un compuesto hecho de un solo sólido.
- Profile. Debe ser un borde cerrado o alambre y debe ser plano.
- Height. Refiere la altura de la extrusión.
- Draft Angle. Un ángulo de inclinación en grados y se activa haciendo clic en:
- Change Direction. La dirección se puede invertir presionando el botón:

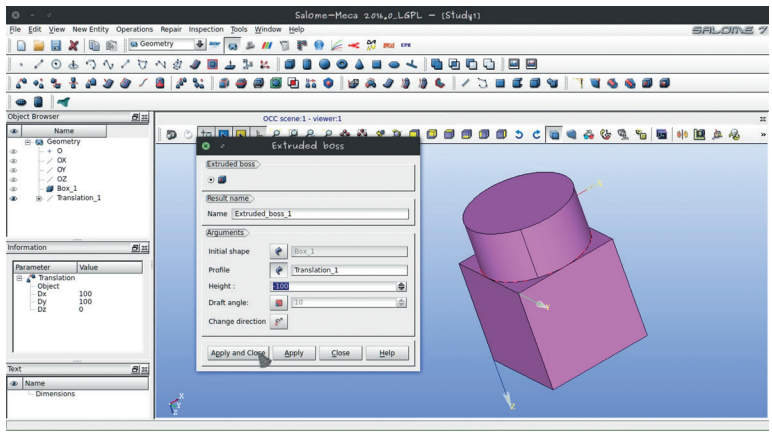


Figura 80. *Extruded Boss*

Chamfer (Chaflán)

Para producir un chaflán, vamos al menú **Operations** > **Chaflán**; esta operación permite hacer chaflán de los bordes de un objeto. Figura 79.

Argumentos:

Name. Es el nombre de la figura resultante.
Main Object. Es el objeto al que se le creará el chaflán.
D. Es la dimensión del chaflán (radio).

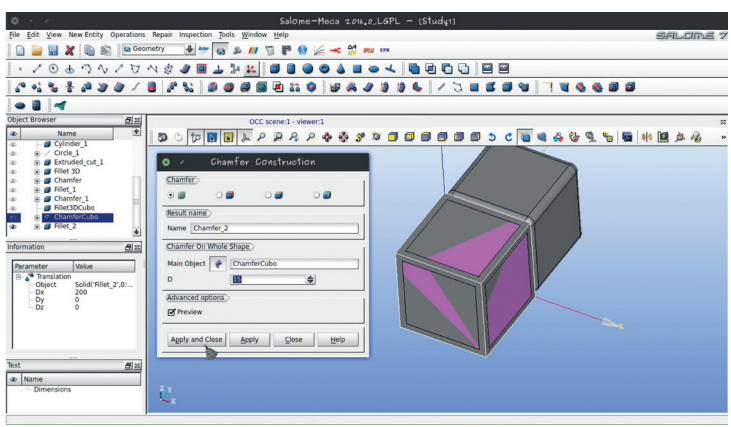


Figura 81. *Chamfer*

Fillet 3D 

Para producir un fillete en 3D vamos al menú **Operations** > **Fillet**; esta operación crea filletes en los bordes de un objeto.

Argumentos:

Main Object. Es un elemento al que se le creará el fillete.
Radius. Es el radio para el fillete. Figura 82.

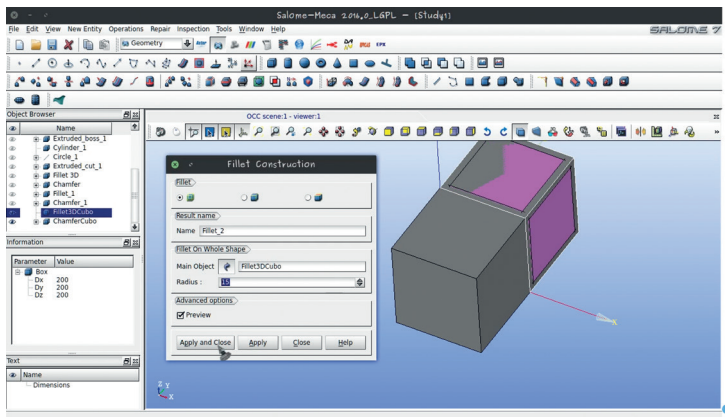


Figura 82. *Radio para el fillet*

La siguiente figura 83 es la vista de un fillete 3D y un chamfer.

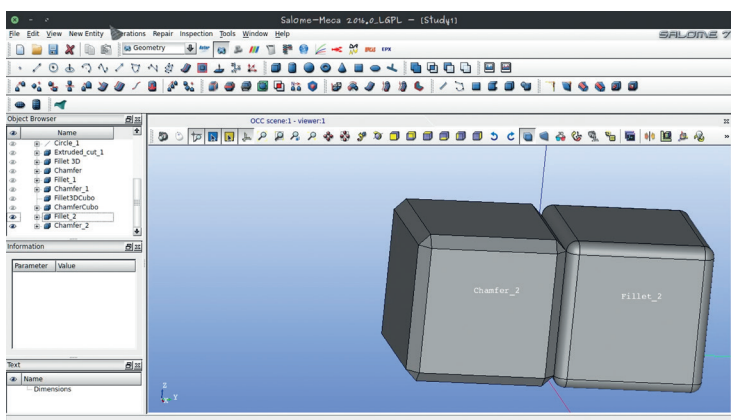


Figura 83. *Fillet 3D y un chamfer*

Geometrías básicas

En todo gemelo digital, la geometría representa un valor medular. Por esa razón, exponemos ahora las herramientas esenciales de la conformación de una estructura básica. Para ello, ofrecemos cinco sencillos ejemplos con distintas formas, tanto planas, curvas, como circulares y volumétricas. Empezaremos con la muestra de un cubo, después expondremos la elaboración de un cilindro, un resorte, una abrazadera y, por último, un tornillo.

1. Creación de un cubo

Creación de un cubo tridimensional con las especificaciones siguientes de Dx: 50, Dy: 300, Dz: 50, Apply o Apply and Close. Figura 84.

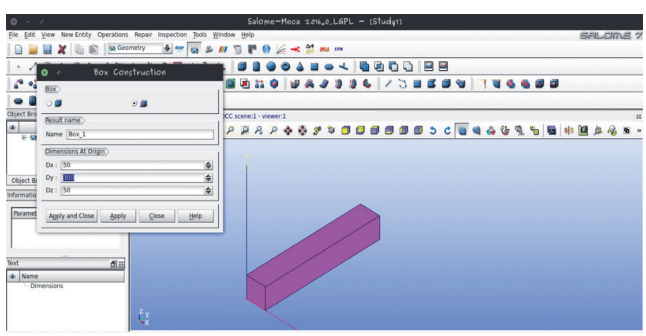


Figura 84. *Creación de un cubo*

Guardamos nuestra pieza en File > Save As como se muestra en la figura 85.

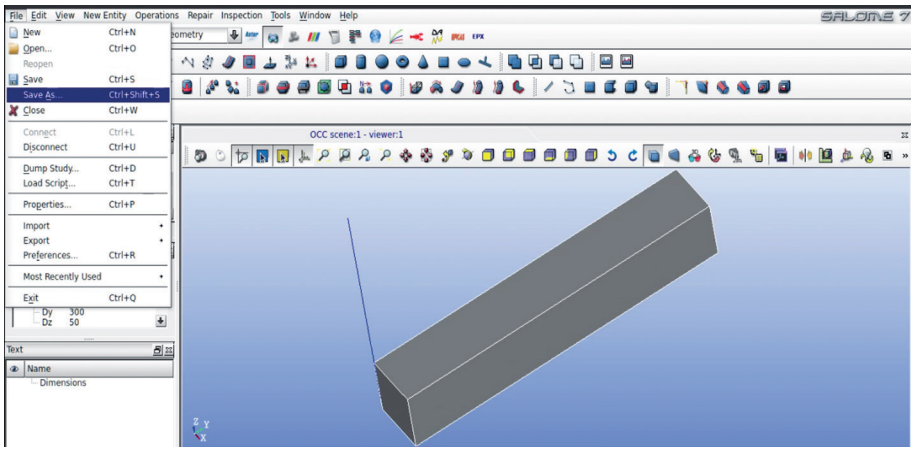


Figura 85. *Pieza guardada*

Lo guardaremos con el nombre de rectángulo en el directorio que deseemos.

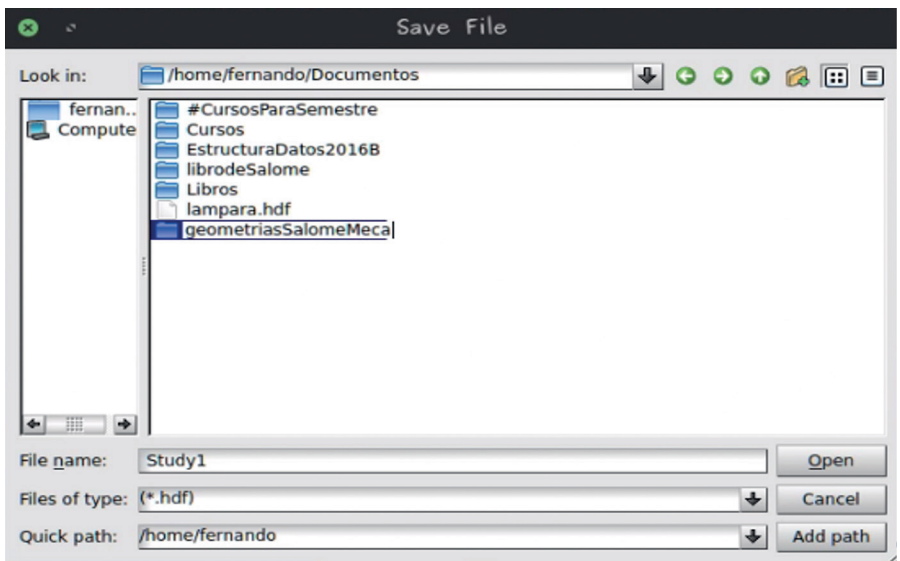


Figura 86. *Pieza nombrada y guardada*

2. Creación de un cilindro

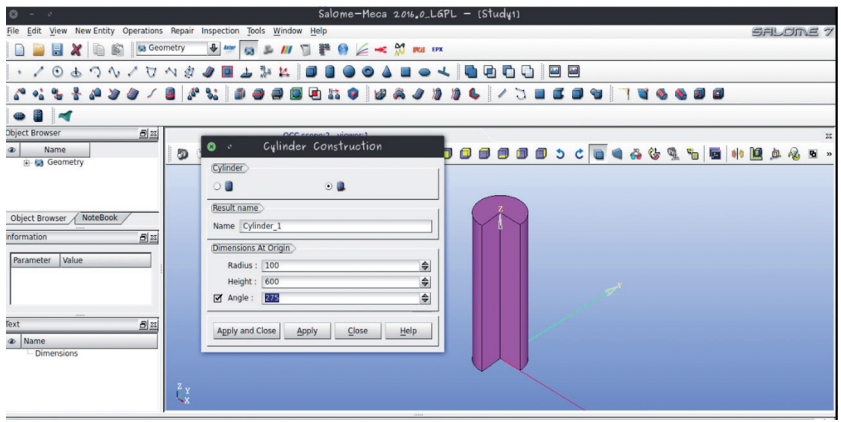


Figura 87. *Creación de un cilindro*

Para comenzar, seleccionamos la casilla Angle que en automático nos dará un cilindro con el ángulo indicado como se muestra en la figura 88.

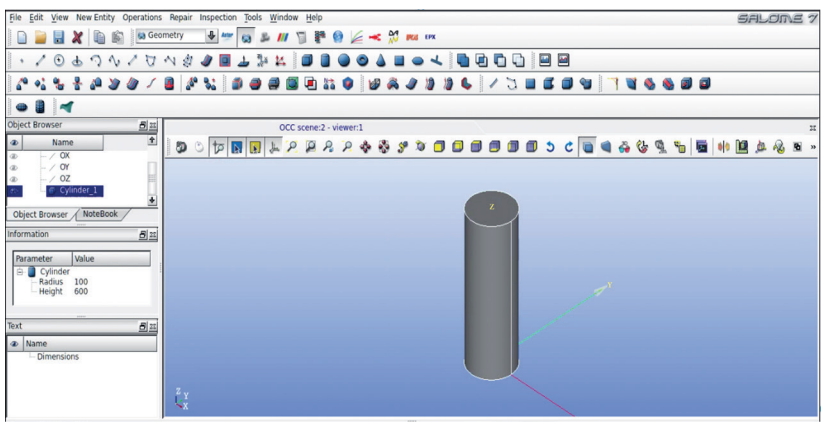


Figura 88. *Cilindro en el ángulo indicado*

En este caso se quedará sin la casilla marcada, aplicamos y cerramos.

Guardaremos nuestra figura **(File > Save As…)** en el directorio que deseemos como lo hicimos anteriormente.

3. Creacion de un resorte (curva)

Nos vamos al ícono de Create a Curve en la barra de herramientas. Figura 89.

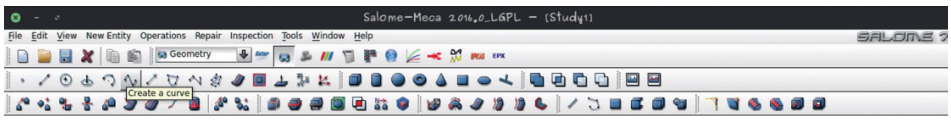


Figura 89. *Identificar Create a Curve*

Seleccionamos la tercera opción de nuestro cuadro de diálogo, elegimos la opción Analytical, los argumentos en esta parte dependerán de la curva o geometría que deseemos obtener. Al finalizar podemos aplicar y cerrar. Figura 90.

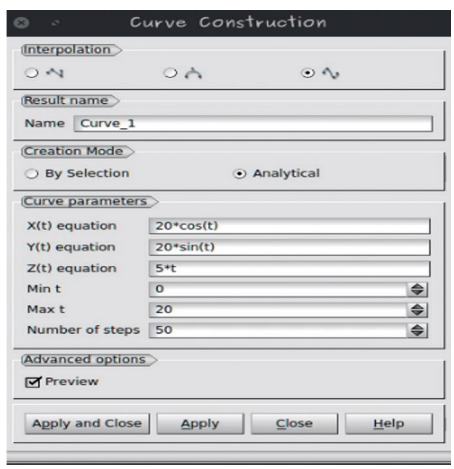


Figura 90. *Seleccionar Analytical*

Posteriormente crearemos un círculo, el cual formará parte del cuerpo del resorte. Figura 91.

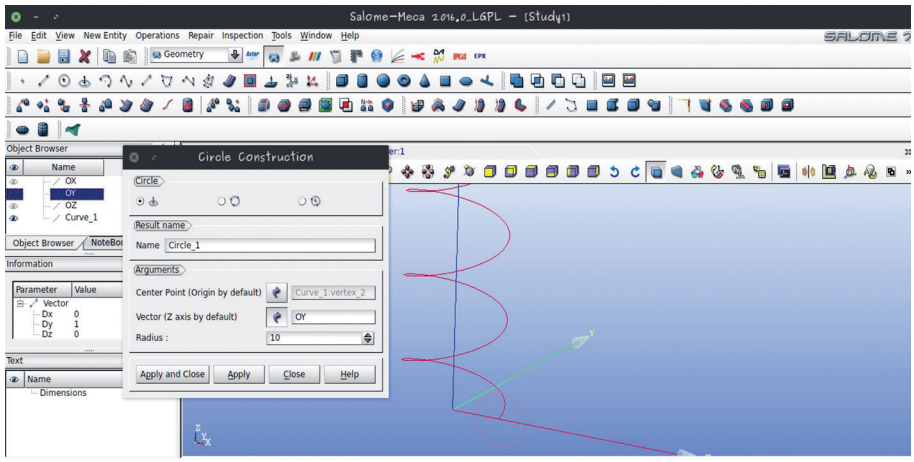


Figura 91. *Creación del círculo*

A nuestro Círculo antes elaborado le crearemos una cara (face), New Entry>build>fase figura 92.

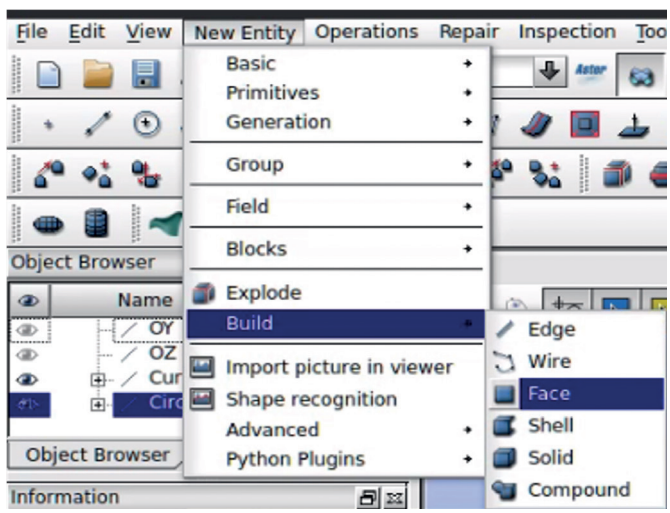


Figura 92. *Crear una cara*

En esta parte sólo elegiremos la geometría para crear la cara que en tal caso será el círculo. Figura 93.

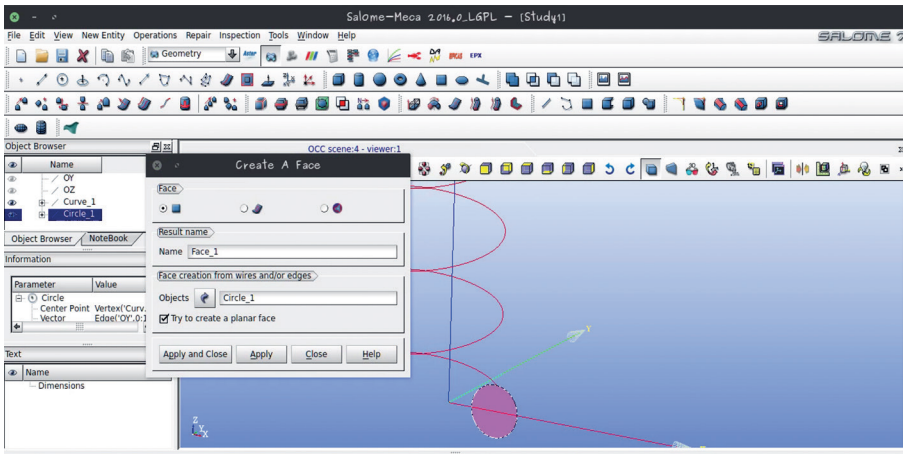


Figura 93. *Elegir la geometría*

Generaremos el cuerpo, creando una extrusión, para ello nos dirigimos al menú New Entity>Generation>Extrusion Along Path. Definir sus argumentos en Base Object elegimos el círculo el cual va indicar el cuerpo, Path Object eligiremos la curva la cual nos indicará la trayectoria y aplicamos y cerramos. Figura 94.

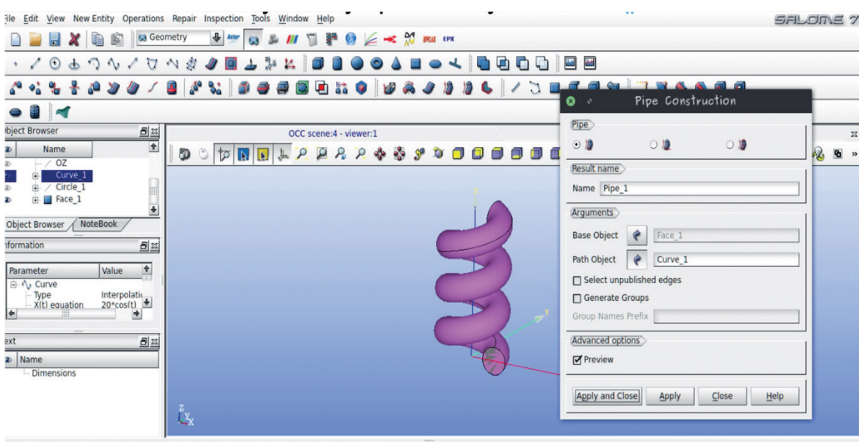


Figura 94. *Creación de extrusión*

Guardamos nuestra figura con el nombre resorte. Figura 95.

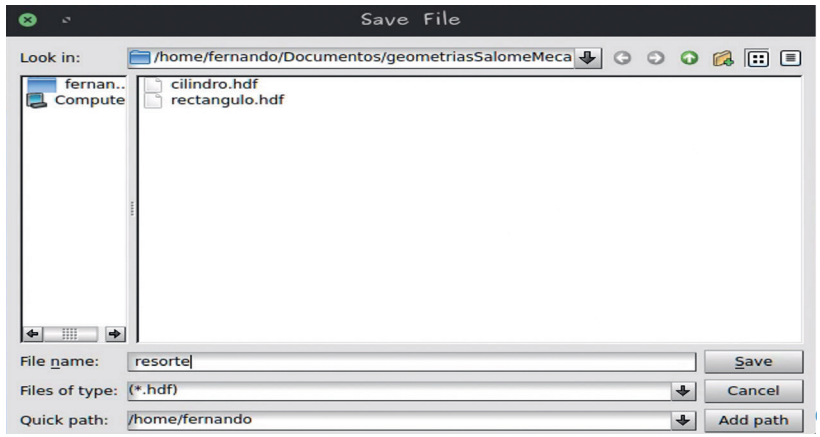


Figura 95. *Guardar como resorte*

4. Creación de una abrazadera

Ahora llevaremos a cabo la creación de una abrazadera y manejaremos las herramientas que nos ofrece el sistema.

Para comenzar construiremos dos cilindros con las siguientes medidas cilindro 1 - Radius 50, heigth 50 y Angle 180. Cilindro 2 - Radius 45, heigth 50 y Angle 180. Figura 96.

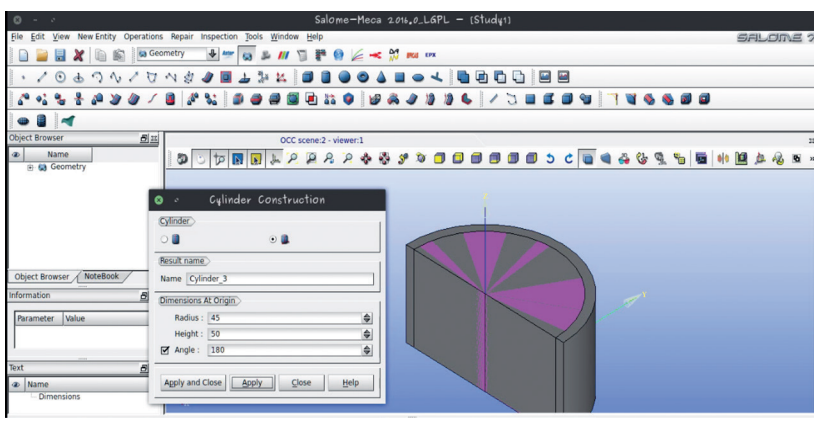


Figura 96. *Creación de una abrazadera*

Al concluir podemos ir al menú Operations>Boolean>Cut, para realizar un corte en la ventana Cut of Objects indicaremos los siguientes argumentos: como nuestro Main Object elegiremos el primer cilindro y como Tool Objects elegiremos el segundo cilindro, y aplicamos y cerramos. Figura 97.

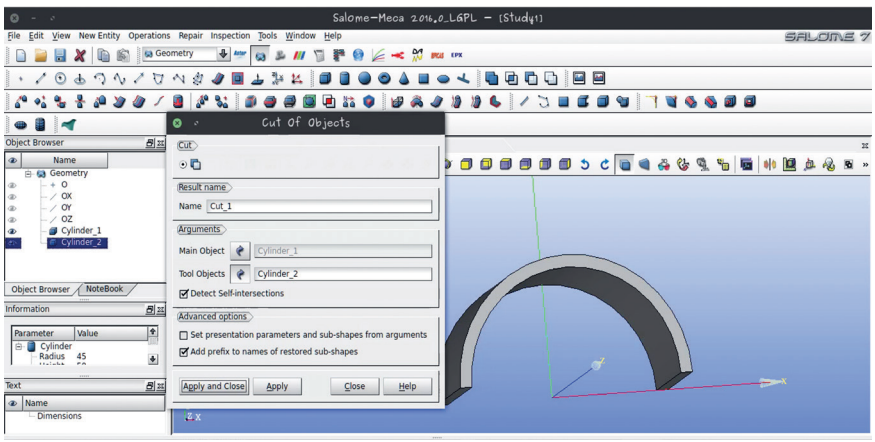


Figura 97. *Cortar objeto*

A continuación, crearemos dos cajas con las siguientes medidas Dx: 10, Dy 5 y Dz 50, las cuales serán las bases de nuestra abrazadera. Figura 98.

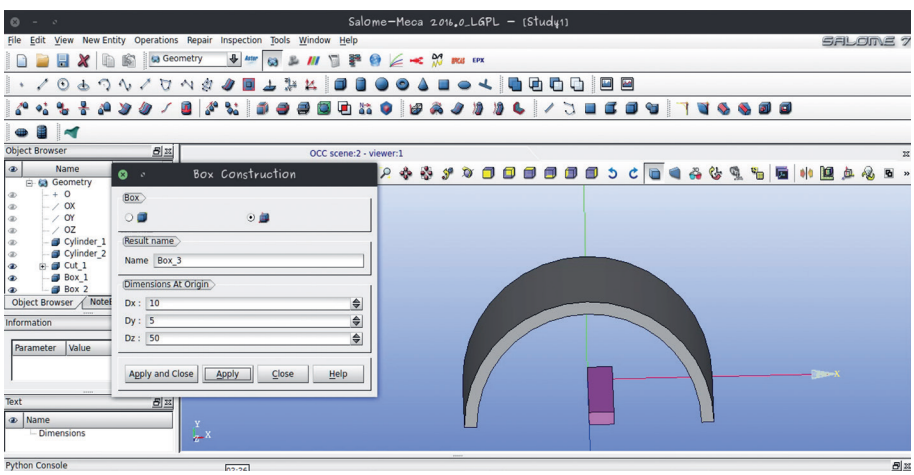


Figura 98. *Creación de dos cajas*

Usaremos el menú Operations>Transformations>Translation para ubicar nuestras cajas a cada extremo de nuestra abrazadera. Figura 98.

Para la caja 1 moveremos sobre la dirección x (Dx) con un valor de 45, para acomodarlo en el extremo derecho.

Para la caja 2 moveremos sobre la dirección x (Dx) con un valor de -45, para acomodarlo en el extremo izquierdo.

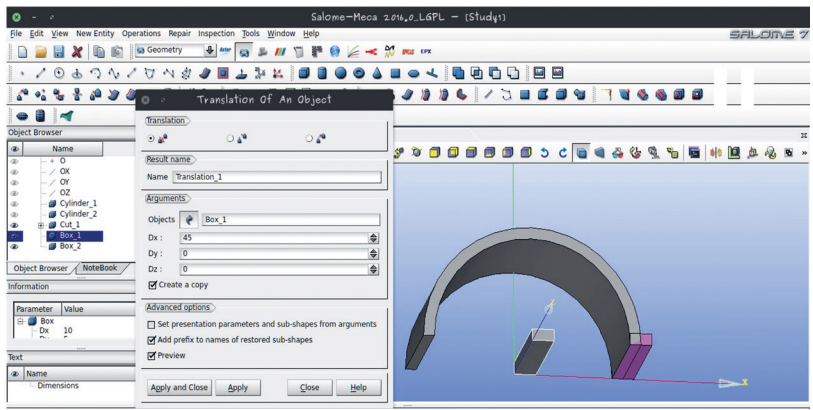


Figura 99. *Aplicando la translación*

Hasta este punto fusionaremos todas las partes para crear un objeto sólido, para ello nos vamos al menú Operations>Boolean>Fuse en la ventana Fuse Objects seleccionamos todos los objetos a fusionar, aplicamos y cerramos. Figura 100.

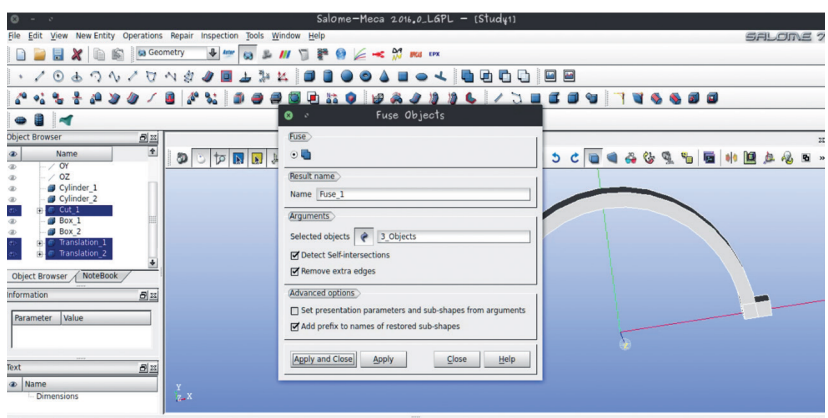


Figura 100. *Creación del objeto sólido*

En este punto nos encontramos con un objeto sólido; para continuar crearemos seis cajas que servirán como base de nuestros tornillos; las dimensiones las podemos observar en la figura siguiente, solo debemos tener en cuenta que estas dimensiones deben ser las mismas para las seis cajas. Cerramos la ventana al finalizar esta acción. Figura 101.

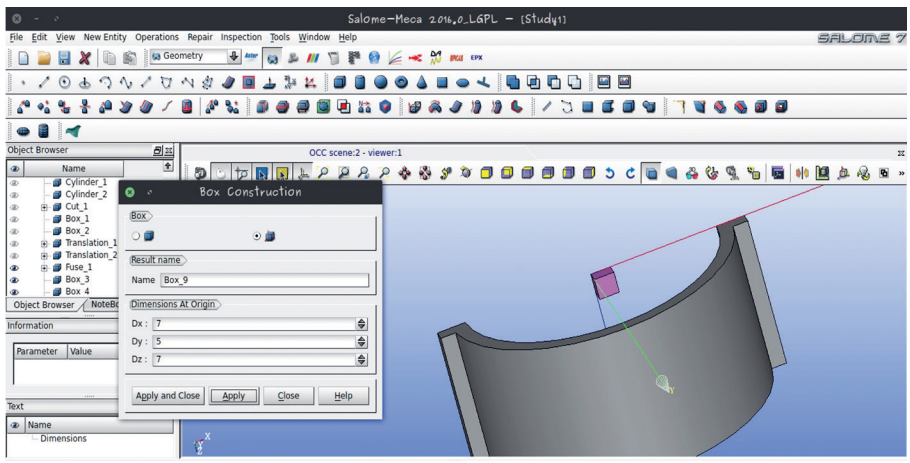


Figura 101. *Base del tornillo*

Movemos cada una de ellas y las acomodaremos en los extremos de nuestra abrazadera, ahora nos vamos al menú Operations>Transformation>Translation.

En la ventana que nos será mostrada deberemos elegir cada objeto que debemos mover (las seis cajas). Los valores de las dimensiones para cada caja figura son los siguientes:

Caja 1 - Dx=48, Dy=5, Dz=0.
Caja 2 - Dx=48, Dy=5, Dz=25.
Caja 3 - Dx=48, Dy=5, Dz=42.
Caja 4 - Dx= -103, Dy= -5, Dz=42.
Caja 5 - Dx= -55, Dy=5, Dz=20.

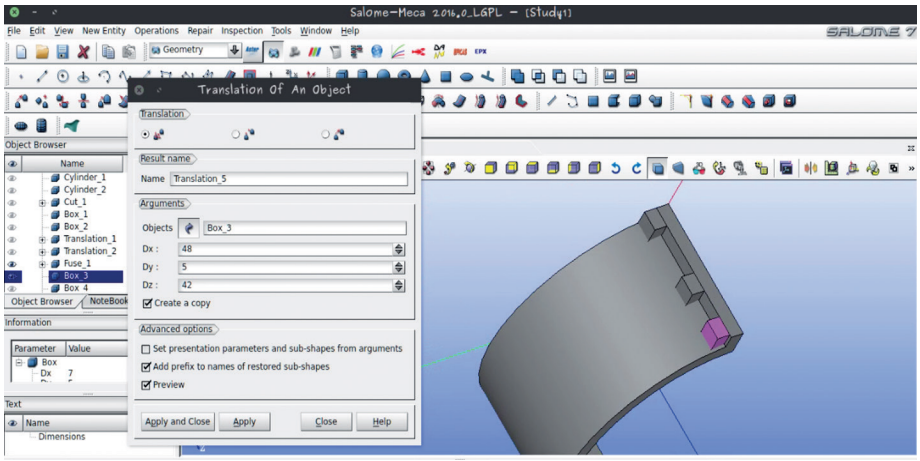


Figura 102. *Acomodar cajas en extremos*

Crearemos un chaflán a estas cajas, por lo que nos dirigimos al menú Operations>Chamfer y en su ventana Chamfer Cronstruction deberemos indicar las dimensiones y argumentos adecuados.

Seleccionando la cuarta opción manejaremos cuatro argumentos: Main Object elegiremos el objeto al cual aplicaremos el chaflán (será cada una de las cajas), Selected Edges deberemos elegir cada uno de los bordes a los cuales será realizado el chaflán (cada borde visible de cada caja), la dimensión será la misma para D1 y D2 con un valor de 2. Figura 103.

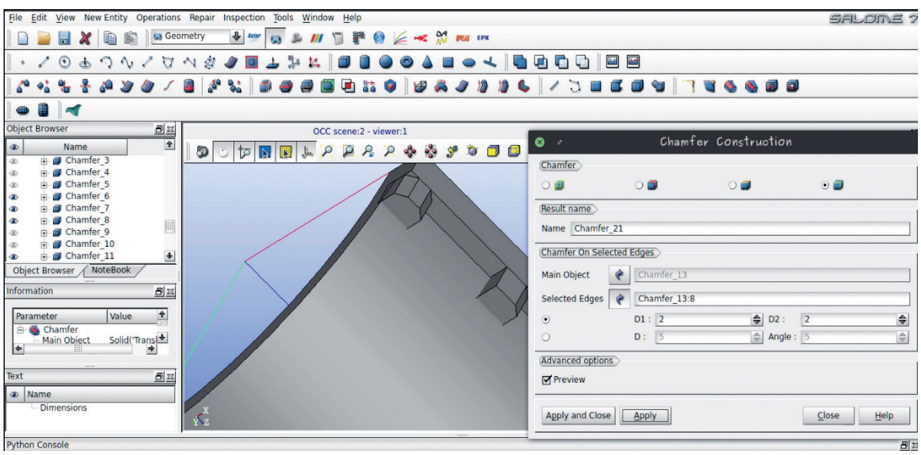


Figura 103. *Construcción de chaflán*

Deberemos crean un objeto sólido hasta este punto para posteriormente girar el objeto. Nos dirigimos a la herramienta Fuse en la cual seleccionaremos todo elemento visible, aplicamos y cerramos. Figura 104.

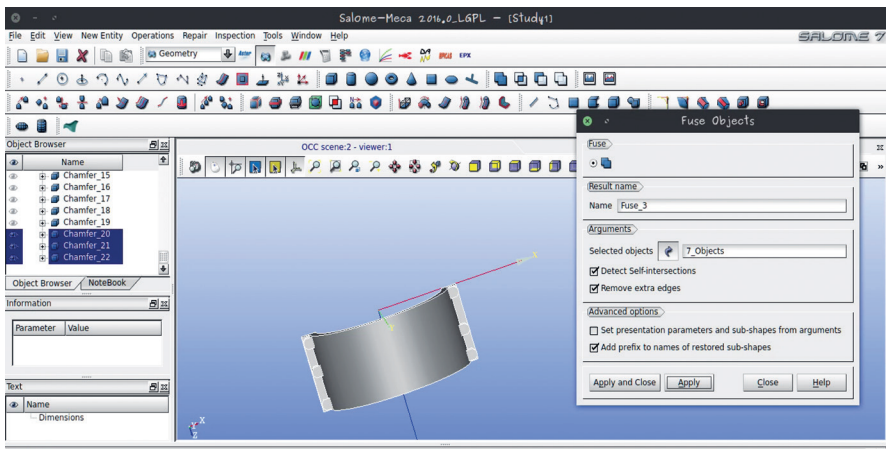


Figura 104. *Creación de objeto sólido*

Se deben crear dos cilindros, el primer cilindro tendrá como radio 7 y longitud 55, el segundo cilindro tendrá como radio 10 y longitud de 6. Aplicamos y cerramos. Figura 105.

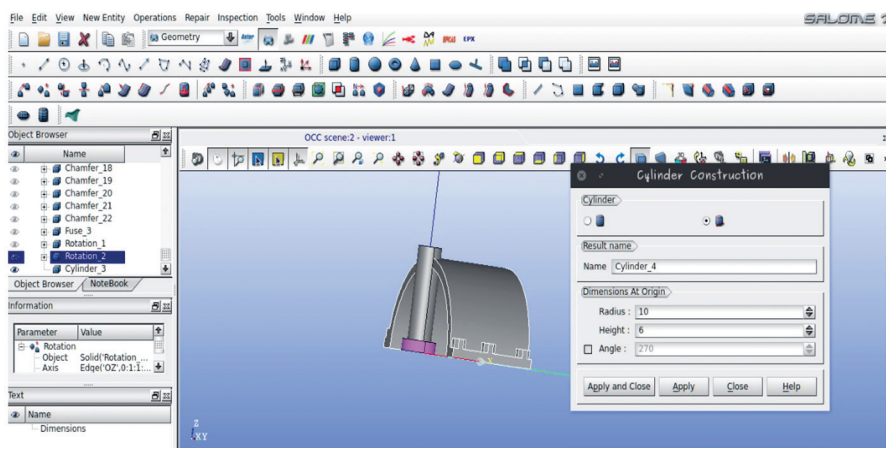


Figura 105. *Dos cilindros creados*

Con el uso de la herramienta Translation moveremos los cilindros en la parte central de nuestra abrazadera. Al cilindro 1 le aplicaremos las dimensiones Dx=0, Dy=25 y Dz=0. Figura 106.

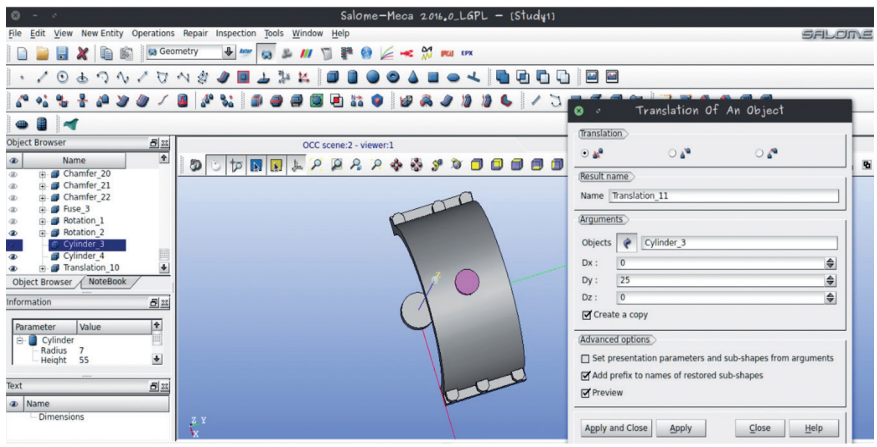


Figura 106. *Aplicar herramienta de translación*

Al cilindro 2 le aplicaremos las dimensiones de Dx=0, Dy=25 Dz=49 y cerramos. Figura 107.

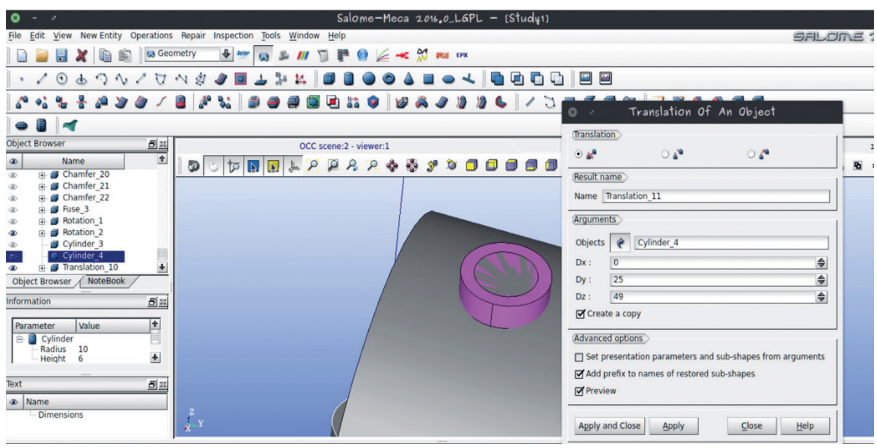


Figura 107. *Cilindro dos, aplicar dimensiones*

Para la creación del orificio de las tuercas dibujaremos un círculo en medio de cada una de ellas. Figura 108.

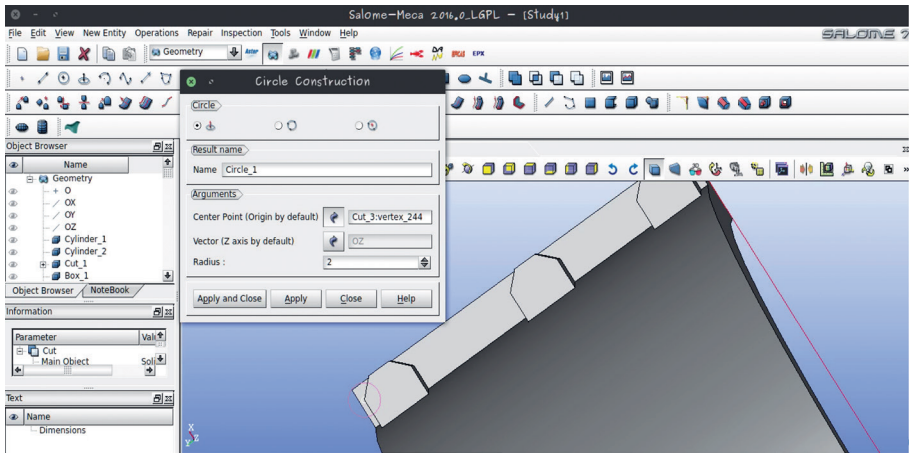


Figura 108. *Orificio para tuercas*

Deberemos trasladarlo hacia el centro como hicimos con el cilindro, para ello utilizaremos la Translation, ingresaremos valores en Dx,Dy y Dz. Aplicaremos esto para cada uno de nuestros círculos creados como lo mostramos en la figura 109.

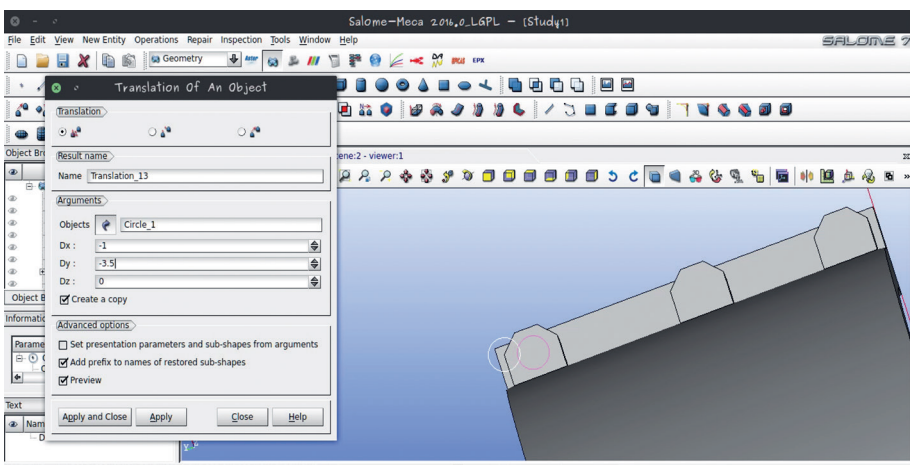


Figura 109. *Posicionarse en el centro*

Crearemos una cara a cada uno de los círculos, como se muestra enseguida.

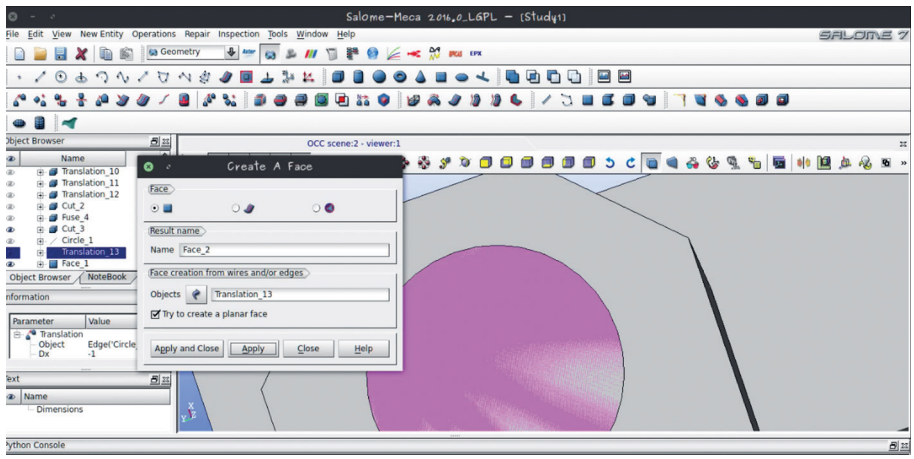


Figura 110. *Crear una cara a los círculos*

Posteriormente realizaremos la extrusión para que el círculo quede vacío. Figura 111.

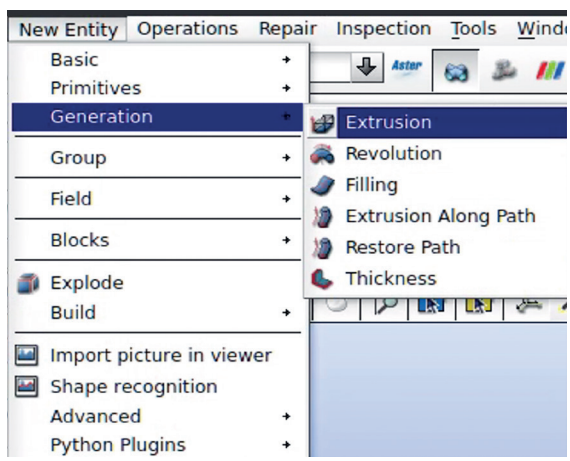


Figura 111. *Extrusión del círculo*

Una vez elegida la extrusión nos aparecerá una ventana llamada (Construction by Extrusion), para el argumento Base deberemos elegir cada uno de los círculos, la dimensión sobre la dirección de z será la misma cuyo valor manejaremos de -9.5, el cual creará el corte necesario. Una vez concluido cerramos nuestra ventana. Figura 112.

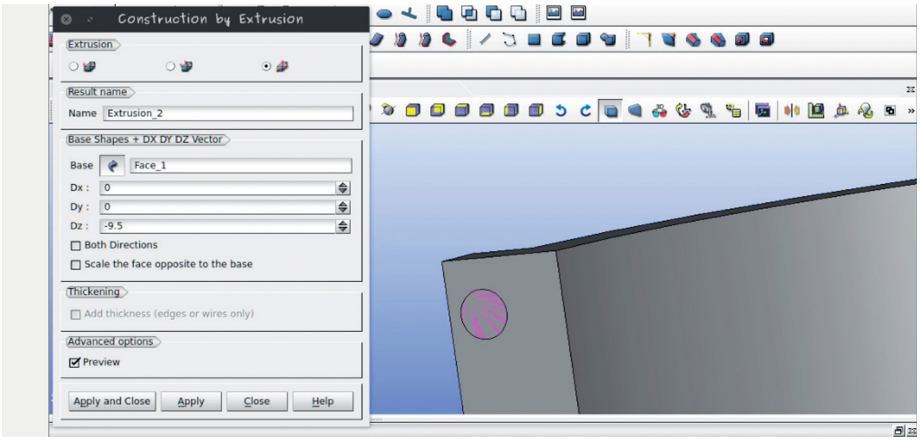


Figura 112. *Construcción de la extrusión*

Realizaremos un corte con la herramienta Cut ubicada en la barra de herramientas o en el menú Operations> Boolean> Cut. Accedemos a ella. Figura 113.

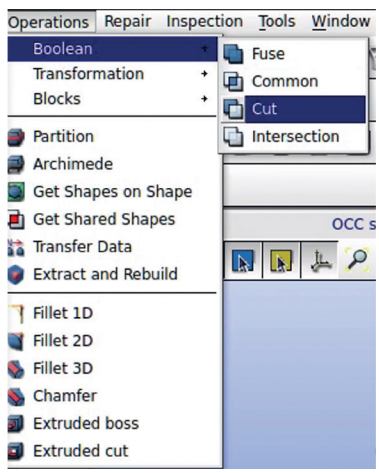


Figura 113. *Herramienta de corte*

Desde nuestra ventana Cut of Objets se mostrarán los argumentos necesarios en los cuales encontramos Main Object, desde aquí seleccionamos el objeto a cortar; en Tool Objects elegiremos el objeto que realizará el corte cada una de las extrusiones realizadas anteriormente. Figura 114.

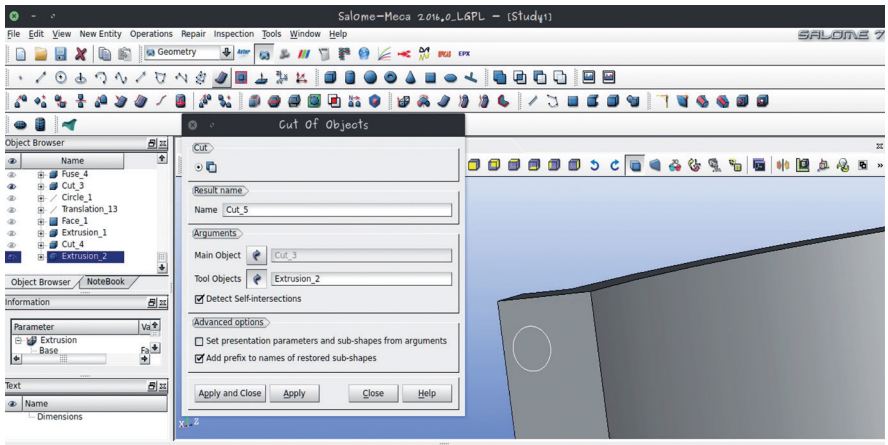


Figura 114. *Cortar objeto*

Para realizar el corte en el centro de nuestra abrazadera primero deberemos fusionar el cilindro más pequeño con nuestra abrazadera. Al ingresar al menú Operations> Boolean> Fuse nos mostrará una ventana cuyos argumentos serán las dos piezas a unir. Figura 115.

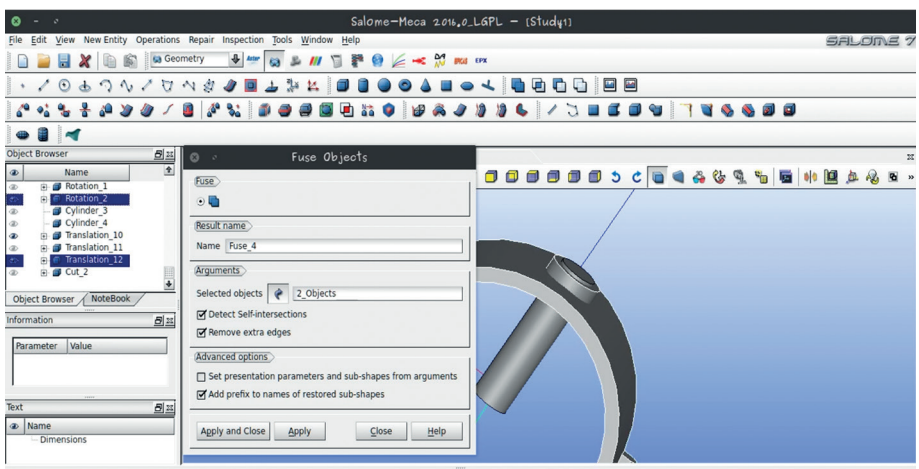


Figura 115. *Fusión del cilindro*

Posteriormente realizaremos un corte a nuestra abrazadera; para ello nos dirigimos al menú Operations> Boolean> Cut, desde donde se mostrará

una ventana como primer argumento (Main Object), se elige el objeto a cortar de la abrazadera; del segundo argumento (Tool Objects) se elige el objeto que realizará el corte en el cilindro, aplicamos y cerramos. Figura 116.

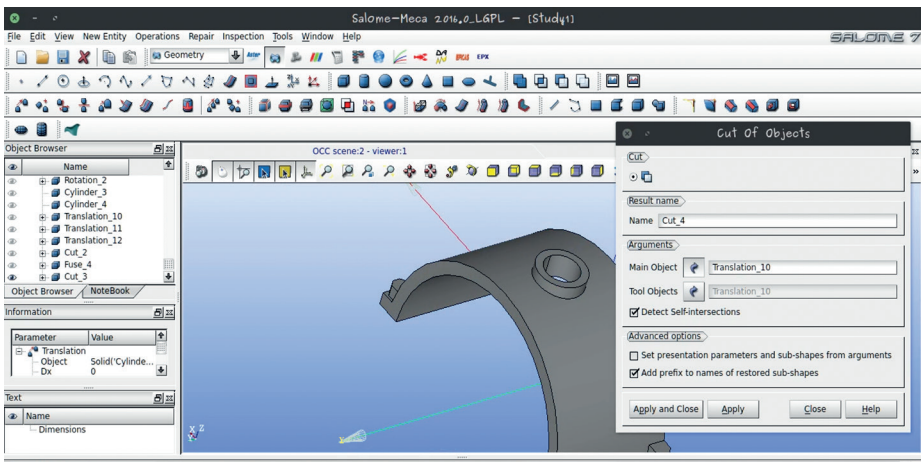


Figura 116. *Realizar corte a la abrazadera*

El resultado hasta este momento se muestra en las figuras 117 y 118.

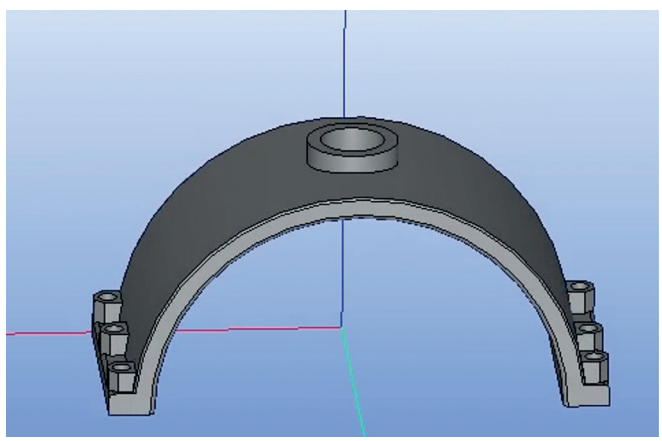


Figura 117. *Resultado vista lateral*

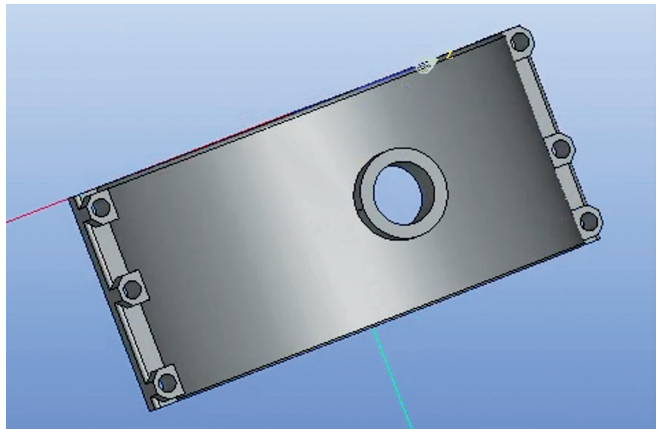


Figura 118. *Resultado vista frontal*

5. Creación de un tornillo

A continuación, llevaremos a cabo la creación de un tornillo con Salome Meca con el diseño que se percibe en la figura 119.

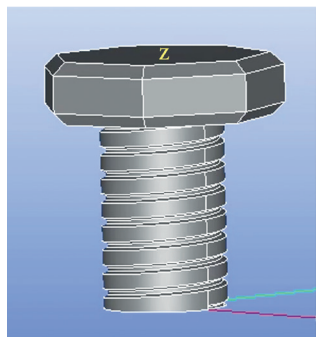


Figura 119. *Tornillo Salome Meca*

Herramienta para el corte

Comenzamos creando un cono con dimensión de .5 de radio y 1 de longitud; de la misma manera creamos un cubo con dimensión .8 sobre eje Z, 2 sobre eje X y Y. Figura 120.

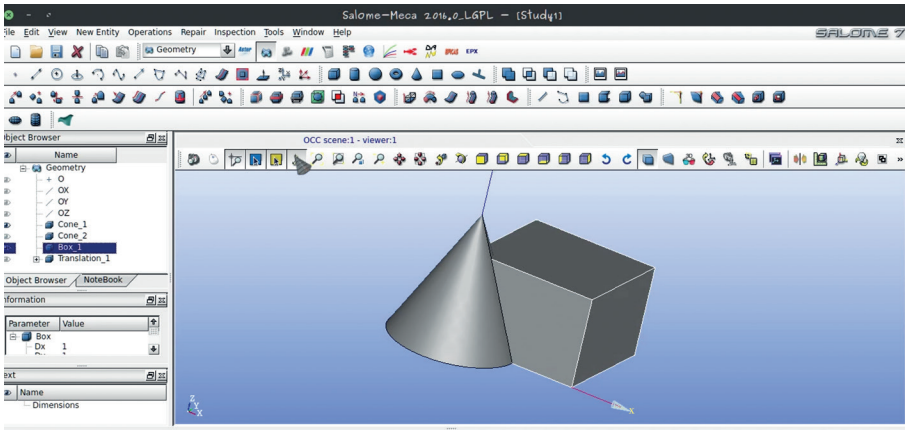


Figura 120. *Creación de cono*

Nos dirigimos al menú Operations – Transformation – Translation para poder mover el cono en medio de nuestro cubo como se muestra en la figura 121.

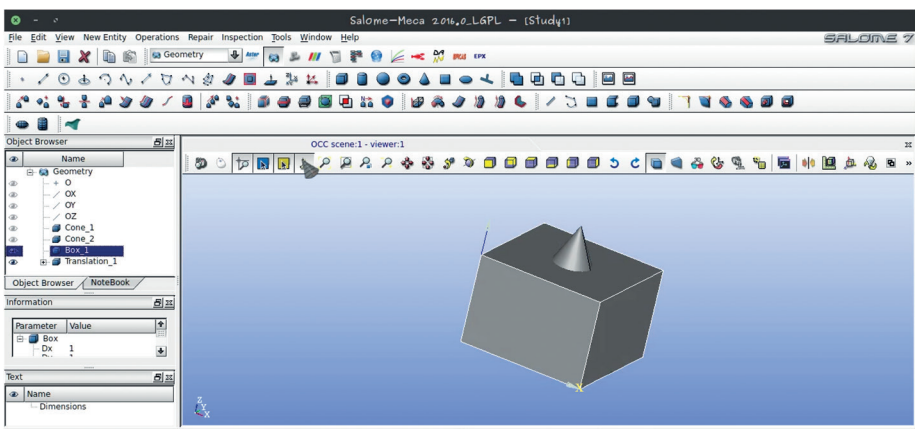


Figura 121. *Cono en medio del cubo*

Una vez reubicado nuestro cono nos dirigimos al menú Operations – Boolean – Common con el cual se realizará un corte de las piezas dando como resultado la parte en la que ambas partes están unidas. Figura 122.

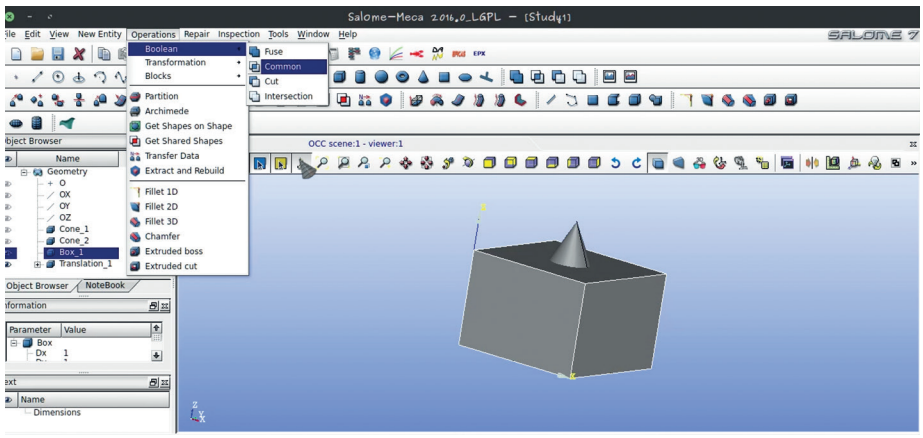


Figura 122. *Corte de las piezas*

A continuación, nos saldrá un cuadro de diálogo que nos indica la ejecución de Common en donde seleccionaremos las piezas a las que queremos realizar la operación. Figura 123.

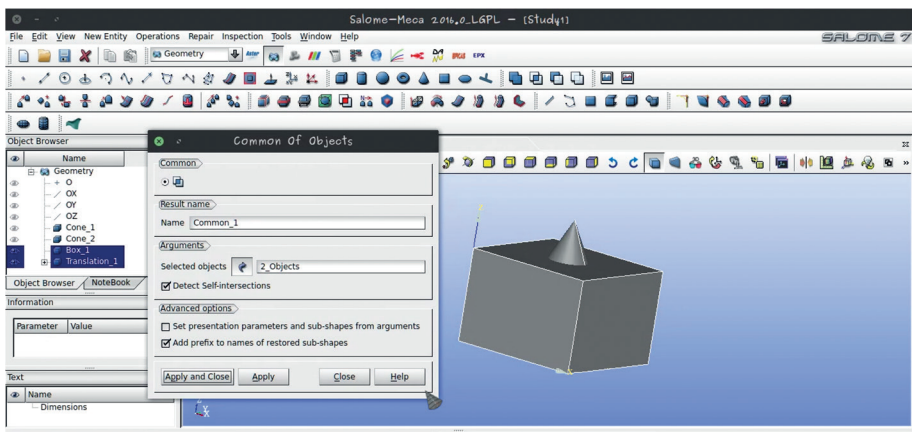


Figura 123. *Ejecución de comandos*

En la parte de Selected objects, podremos observar que indica 2_Objetos, de manera que podemos aplicar y cerrar.

El resultado de esta operación la podremos observar en la figura 124 en forma de cono con la punta eliminada, una vez obtenido este resultado continuaremos con el proceso.

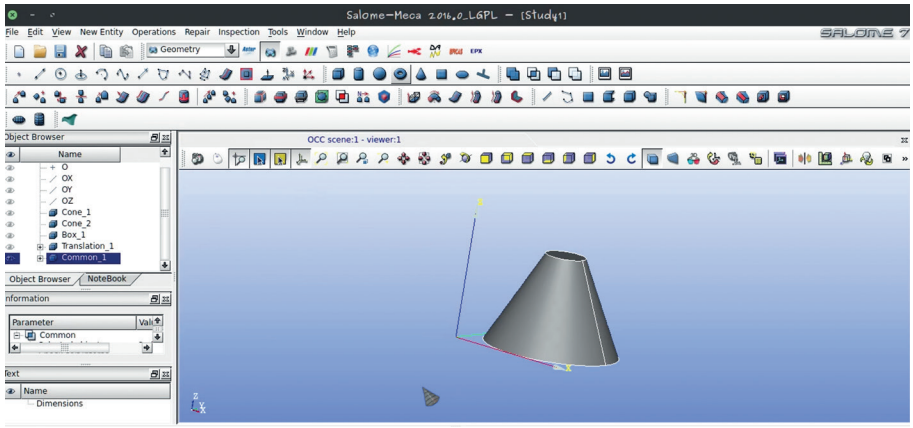


Figura 124. *Resultado de operación*

Crearemos un cuadro con dimensiones de Height y Width de 2 cm, le daremos una orientación de OYZ, podemos dar aplicar y cerrar. Figura 125.

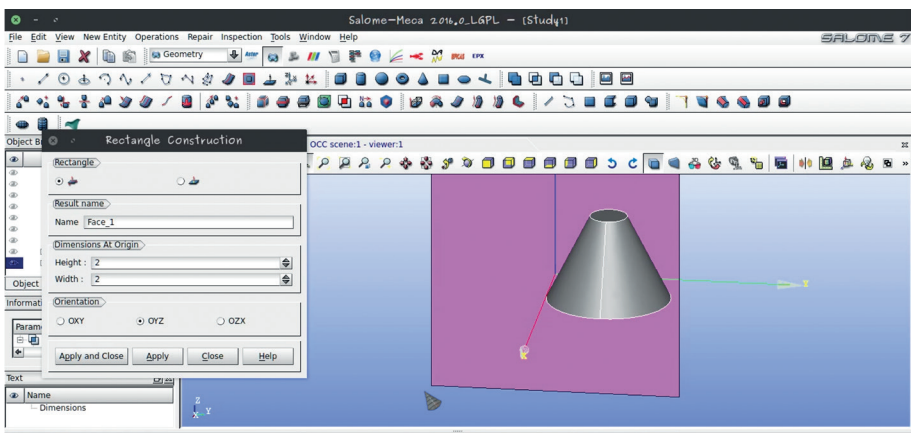


Figura 125. *Cuadro con dimensiones*

Realizamos la operación Translation para mover nuestro cuadro creado en la parte media de nuestro cono cortado. Figura 126.

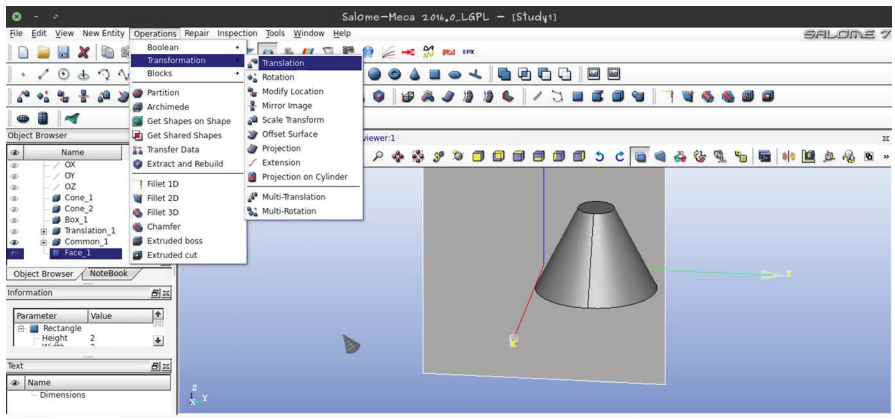


Figura 126. *Aplicar translación*

Nos abrirá su respectivo cuadro de diálogo, en la parte de Objects elegimos la parte a mover; en este caso será el cuadro. Más abajo encontramos en qué dirección queremos moverla (Dx, Dy, Dz). En particular, la moveremos en el eje x solo .5, aplicamos y cerramos. Figura 127.

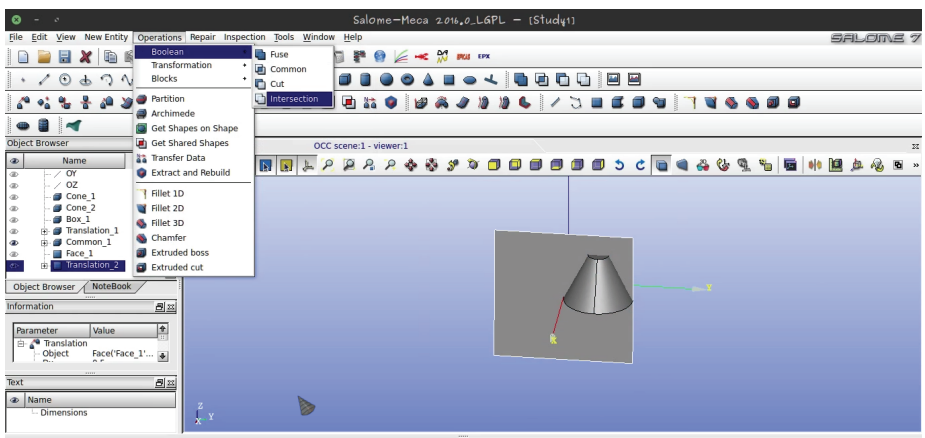


Figura 127. *Cuadro de diálogo*

Nos dirigimos al menú Operations – Boolean – Intersection, esta operación dibuja todas las partes de nuestras figuras donde intersectan.

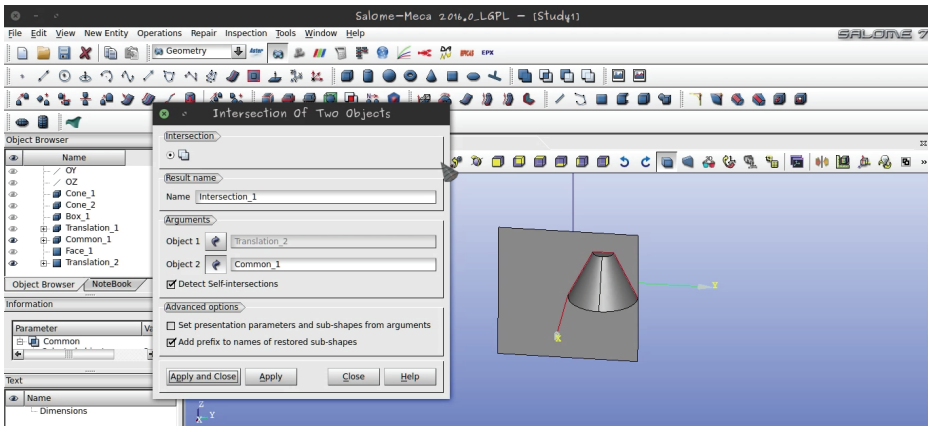


Figura 128. *Aplicar intersección*

En la parte argumentos (Arguments) tenemos object 1 y 2, en los cuales queremos realizar dicha operación, en este caso será el cuadro y el cono. Figura 129.

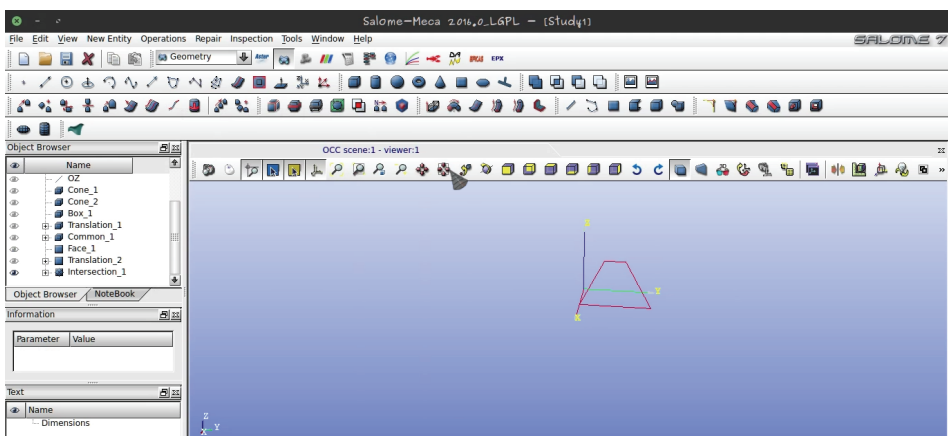


Figura 129. *Argumentos*

Nuestro resultado de la operación intersección será un trapecio, como se muestra en la figura 130.

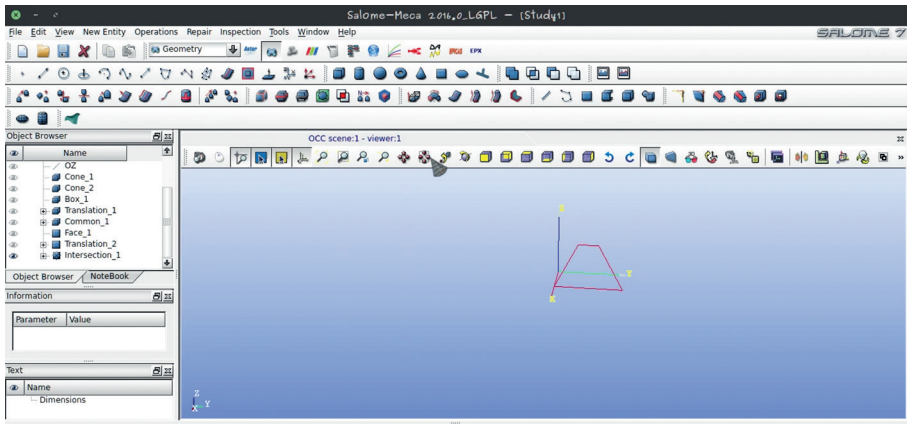


Figura 130. *Resultado obtenido: trapecio*

Ahora nos dirigimos al menú New Entity – Build – Face para crear una cara a nuestro trapecio obtenido anteriormente. Figura 131.

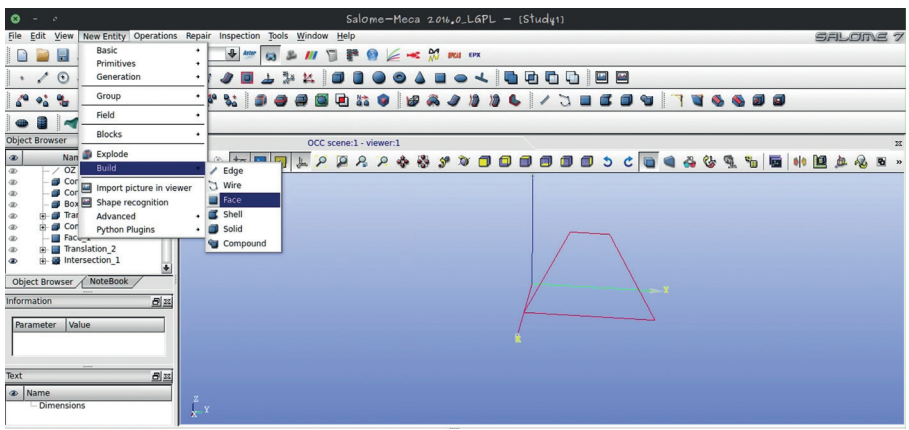


Figura 131. *Crear una cara*

Aparecerá en pantalla su respectivo cuadro de diálogo en donde a la parte de Objects le indicaremos a qué figura queremos darle una cara, como lo mostramos en la figura 132.

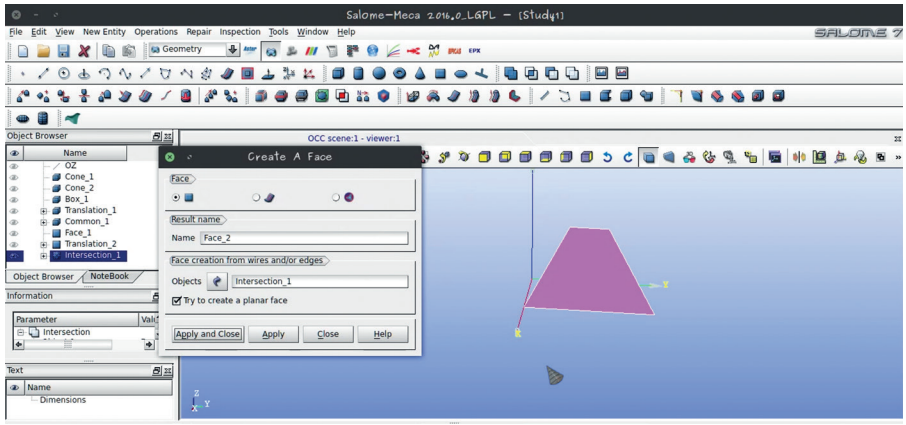


Figura 132. *Indicar la figura*

De momento no usaremos la figura creada (forma de trapecio), ya que esta será utilizada para crear un corte; podemos ocultarla para no confundirnos, lo podemos realizar dando Clic en el ojito que se encuentra en Object Browser, ubicado en la parte izquierda del programa. Figura 133.

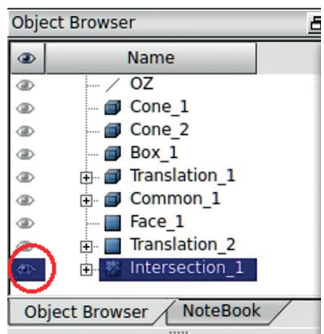


Figura 133. *Ocultar el trapecio*

Una vez ocultado todo para no tener confusiones seguiremos con la creación del cuerpo de nuestro tornillo.

Cuerpo del tornillo

Utilizando la herramienta cilindro (Cylinder), en su cuadro de diálogo veremos la parte Dimensions At Origin, tendremos tres apartados Radius

(radio), Height (altura), Angle (ángulo), mediante los cuales utilizamos un radio 5 cm y altura de 20 cm. Aplicamos y cerramos. Figura 134.

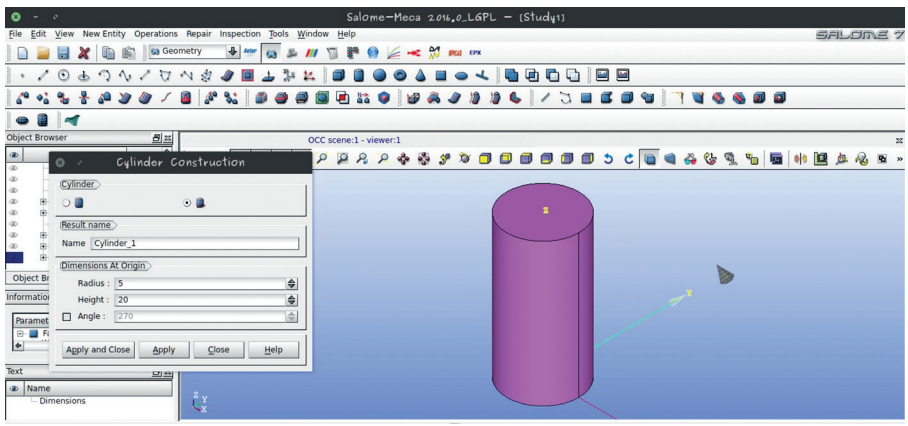


Figura 134. *Creación del cilindro*

Quitamos de la vista el cilindro y proseguimos, ahora crearemos un vector (ícono del vector encerrado en un círculo azul), en su cuadro de diálogo elegimos la segunda opción encerrado en un círculo rojo, más abajo encontramos el apartado Coordinates, le damos una longitud de 20 sobre el eje Z. Aplicamos y cerramos, de igual forma lo ocultamos y continuamos.

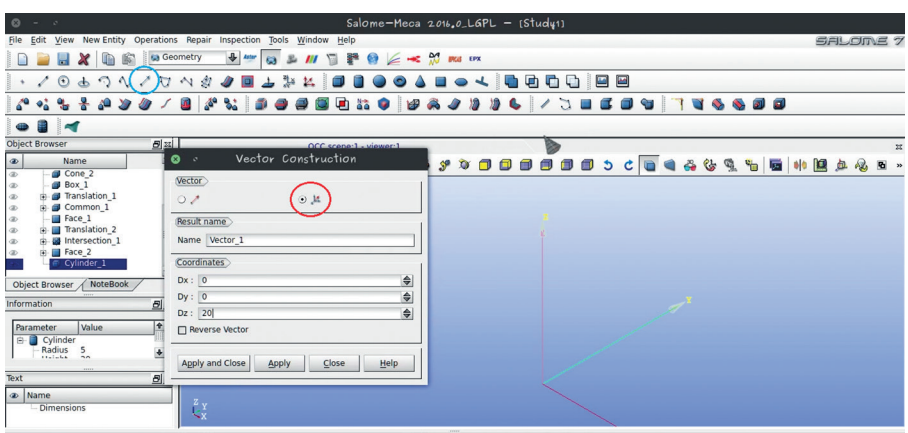


Figura 135. *Crear vector*

Visualizamos el cilindro, crearemos una curva, para ello le damos clic en el ícono de curva (encerrado en un círculo azul), en su cuadro de diálogo, elegimos la tercera opción encerrada en un círculo rojo, en el apartado Curve parameters, le damos los parámetros para nuestra curva como se muestra en la figura de abajo. Aplicamos y cerramos. Figura 136.

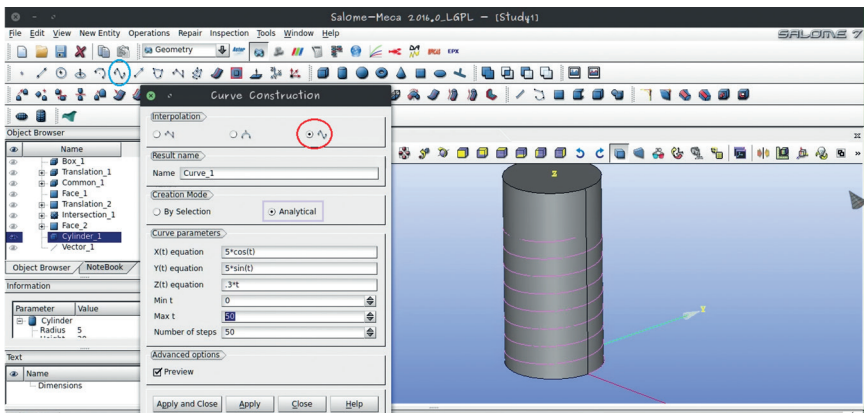


Figura 136. *Crear curva en el cuerpo*

Ahora realizaremos una rotación al trapecio; para ello deberemos visualizar la cuerda y el trapecio, solo esas dos partes visualizadas, lo demás queda oculto para facilitar las siguientes operaciones. Nos dirigimos a menú Operations – Transformation – Rotation.

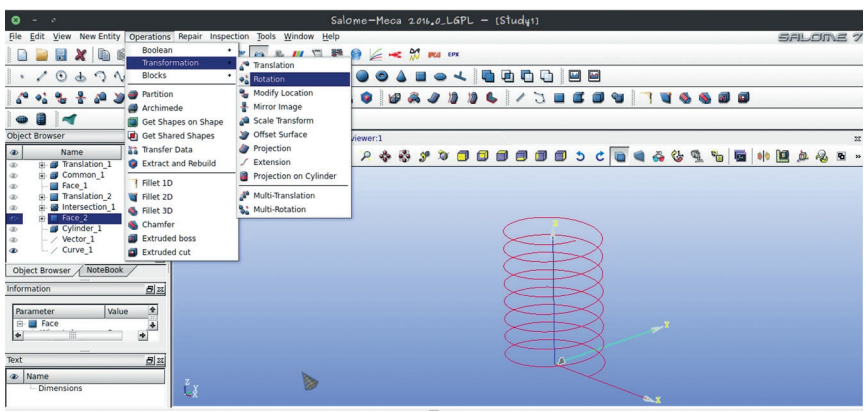


Figura 137. *Creación de la cuerda al tornillo*

En los argumentos de la ventana de Rotation of Object nos encontramos Objects en esta parte vamos a elegir qué objeto vamos a rotar (trapecio), en la parte de Axis vamos a elegir sobre qué eje va a rotar, en este caso será sobre el eje OZ, en la parte de Angle (ángulo) vamos a elegir cuántos grados va a rotar el objeto hasta la posición deseada. Aplicamos sin salir. Figura 138.

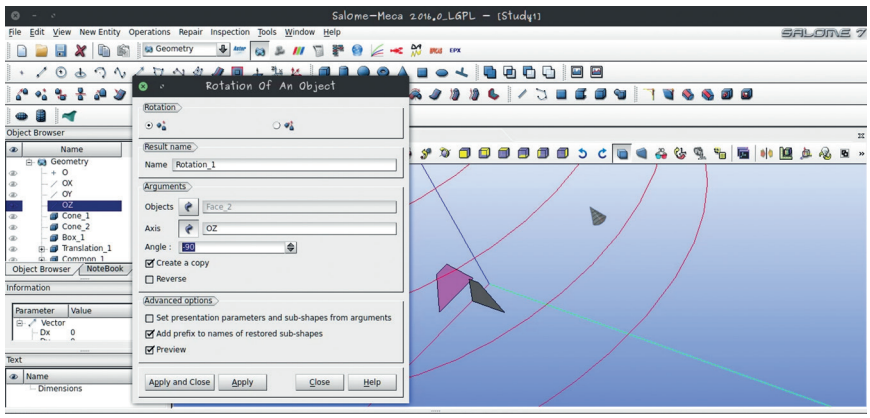


Figura 138. *Rotación de trapecio eje OZ*

De la misma manera, ahora giramos sobre el eje OY -90 grados, de esta forma la parte más pequeña de nuestro trapecio quedará hacia la parte negativa de nuestro eje X terminamos con la rotación, podemos aplicar y cerrar. Figura 139.

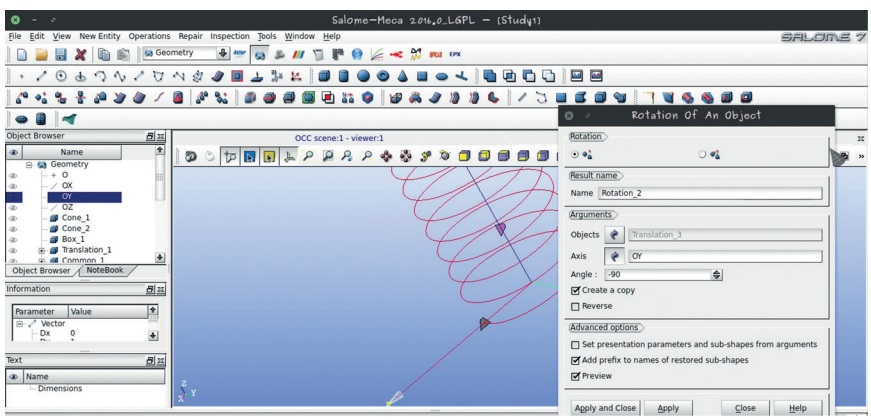


Figura 139. *Rotación de trapecio eje OY*

Tenemos nuestro objeto en la posición de deseada pero no en el punto deseado, para ello nos dirigimos al menú Operations – Transformation – Translation para poder mover nuestro objeto. Figura 140.

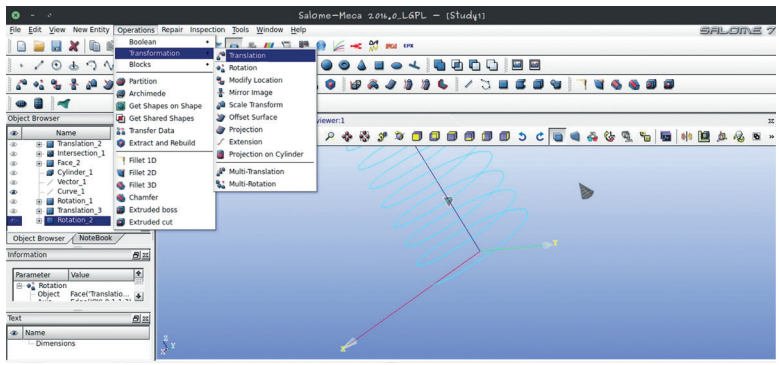


Figura 140. *Aplicar translación*

En su respectiva ventana Translation of an Object nos ubicamos en los Arguments (argumentos), los cuales, si observamos la figura 141, solo nos indica las direcciones en las cuales moverlos le aplicaremos valor a Dx = 5 y a Dz = -5, para ubicar a nuestro trapecio de la parte central al inicio de nuestra cuerda, mostrado en la figura siguiente. Tener en cuenta que los valores podrán ser diferentes en los cuales pueden ir probando en las direcciones que nos ofrece para poner nuestro objeto en la parte deseada. Una vez que nuestro objeto se encuentre en donde lo necesitamos podemos aplicar y cerrar.

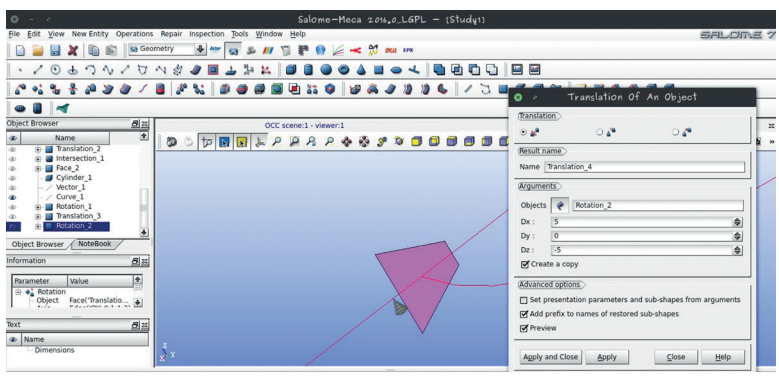


Figura 141. *Direcciones de argumentos*

Al tener ubicado nuestro trapecio y nuestra cuerda creada correctamente, los cuales van hacer nuestro objeto de corte, también hay que tener visualizados el cilindro y el vector; a continuación, podemos proseguir al siguiente paso.

Vamos a crear nuestro objeto de corte, para ello nos vamos al menú New Entity – Generation – Extrusion Along Path. Figura 142.

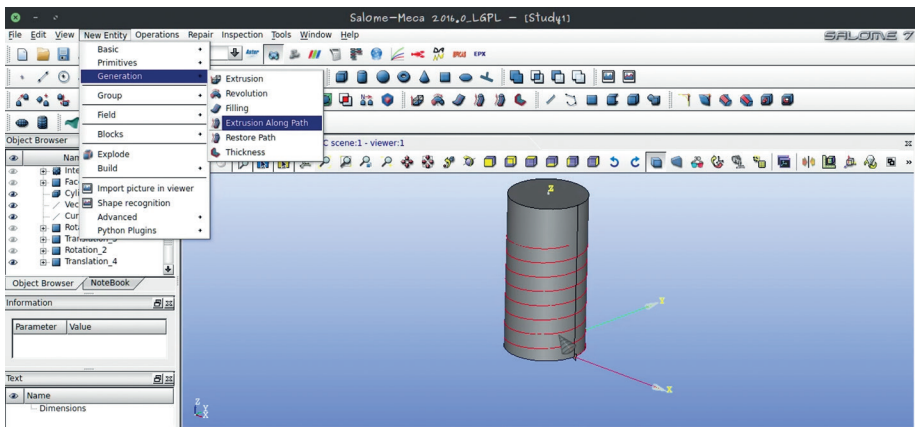


Figura 142. *Objeto de corte*

Nos mostrará su respectiva ventana Pipe Construction desde la que elegiremos la segunda opción encerrada en un círculo rojo en la parte de los Arguments. Sobre Base Object indicaremos el objeto al que deseamos crear la extracción, en este caso es al trapecio; en Path Object indicaremos la parte como trayectoria que requerimos aplicar, en este caso sería la cuerda. Por último, la opción BiNormal es para indicar una orientación, elegimos nuestro vector anteriormente creado, una vez indicados nuestros argumentos podemos aplicar y cerrar.

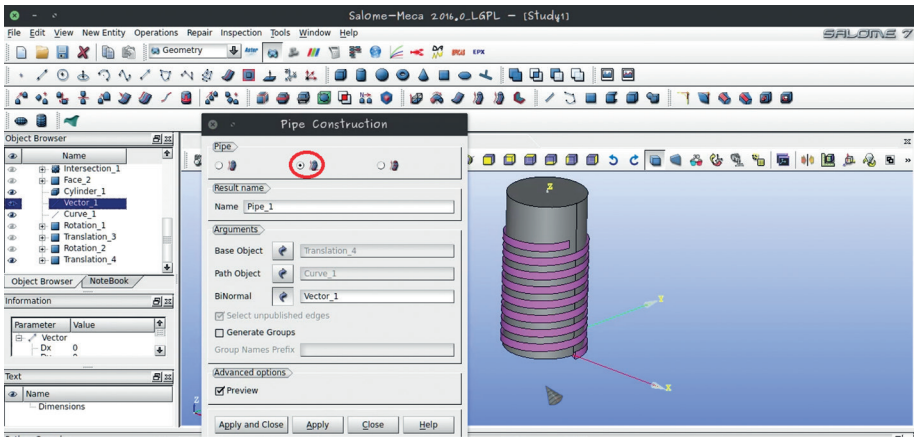


Figura 143. *Extrusión del trapecio*

El siguiente paso será realizar un corte con nuestra extrusión creada anteriormente nos dirigimos al menú Operations – Boolean – Cut, con este realizaremos el corte más tarde. Figura 144.

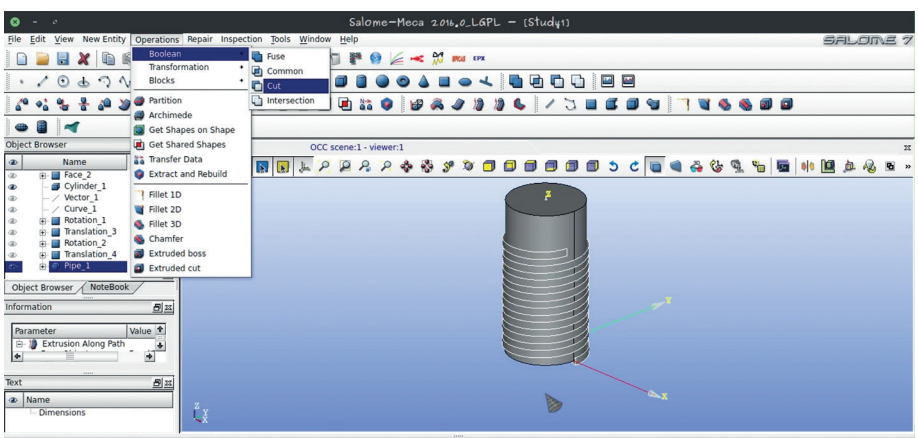


Figura 144. *Corte a extrusión*

Nos mostrará su ventana Cut of Objects; en el apartado de argumentos encontraremos Main Object desde donde indicaremos la pieza u objeto al que queremos realizar el corte. Nosotros elegimos el cilindro, enseguida encontramos la categoría Tool object para indicar la pieza u objeto que

realizará el corte, es decir, la extracción que hemos creado. Concluidos estos argumentos necesarios podemos acceder, aplicar y cerrar. Figura 144.

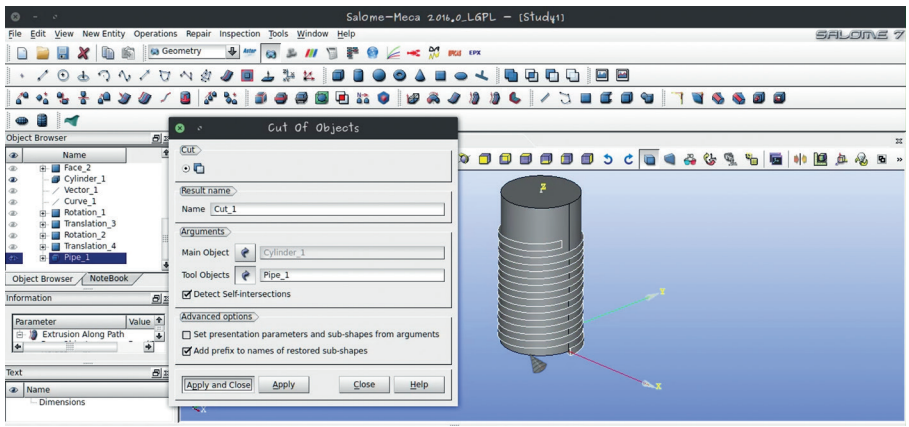


Figura 145. *Indica la pieza*

Al término de todas las operaciones realizadas anteriormente, obtendremos el resultado en la figura 145, así podremos continuar con la creación de nuestro tornillo.

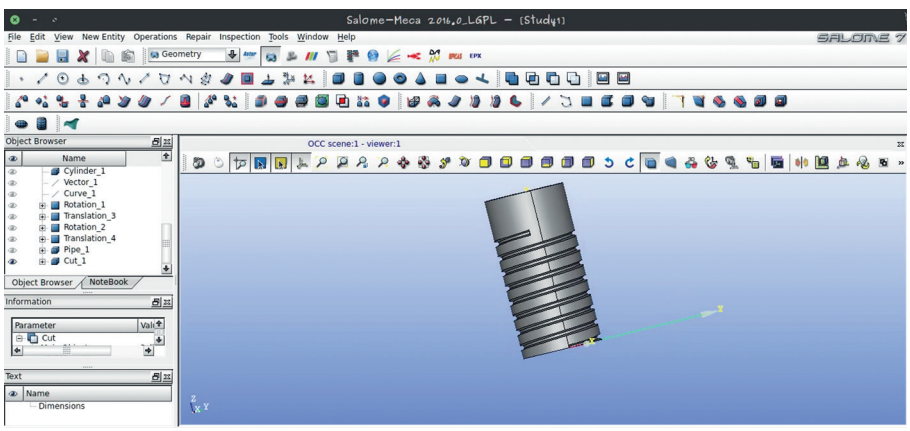


Figura 146. *Resultado de operaciones*

En esta parte llevaremos a cabo la construcción de la cabeza de nuestro tornillo.

Cabeza del tornillo

Como primer paso con la ayuda de la herramienta Disk (disco), en nuestro argumento aplicaremos un diámetro de 9 cm. Con una orientación por defecto y aplicamos sin cerrar (clic Apply). Figura 146.

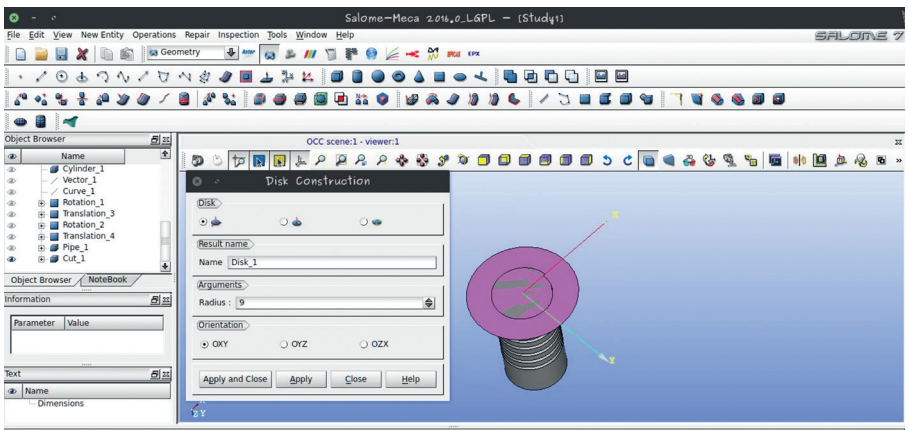


Figura 147. *Crear disco 1*

Crearemos un nuevo disco, usando un diámetro de 4 cm con una OXY; una vez concluido, aplicamos y cerramos. Figura 147.

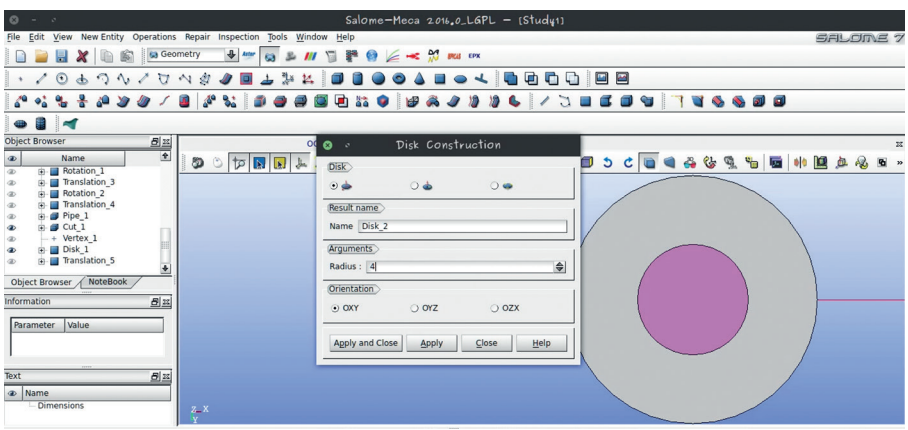


Figura 148. *Crear disco 2*

Haciendo uso de la herramienta Translation Operations – Transformation – Translation, moveremos sobre el eje X nuestro disco más pequeño, si apreciamos la figura, enseguida se encuentra una flecha (roja), en la parte izquierda se encuentra el contorno del disco en su posición original, en la parte derecha un círculo rosa; la flecha indica una intersección que nos advierte la posición adecuada de la nueva posición de nuestro disco. Se debe tener en cuenta que esta posición puede variar dependiendo de lo que se necesite. Para nuestro proyecto específico se realizó de esta manera. Una vez en la posición deseada, aplicamos y cerramos. Figura 148.

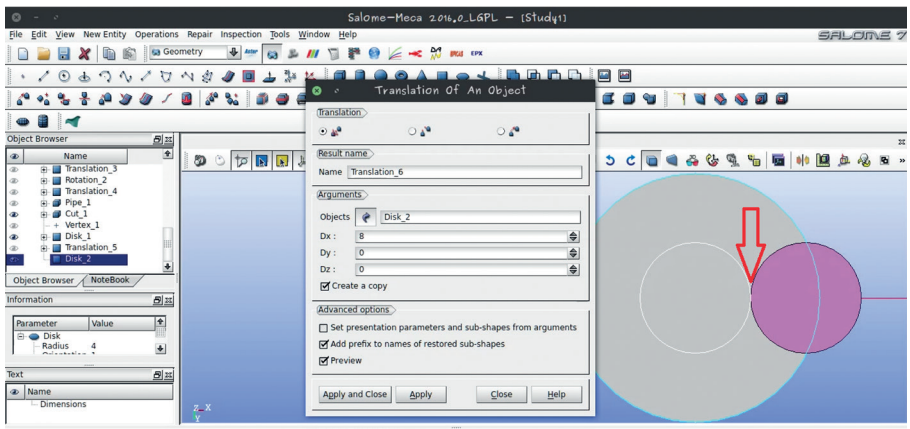


Figura 149. *Mover en el eje X*

A continuación, multiplicamos el disco pequeño para formar un heptágono; para ello vamos al menú Operations – Transformation – Multa-Rotation; en su ventana de operaciones elegiremos la opción predeterminada en la parte Main Object elegimos el objeto a multi-rotar sobre Nb. Times, será el número de veces que lo rotaremos. Aplicamos y cerramos. Figura 149.

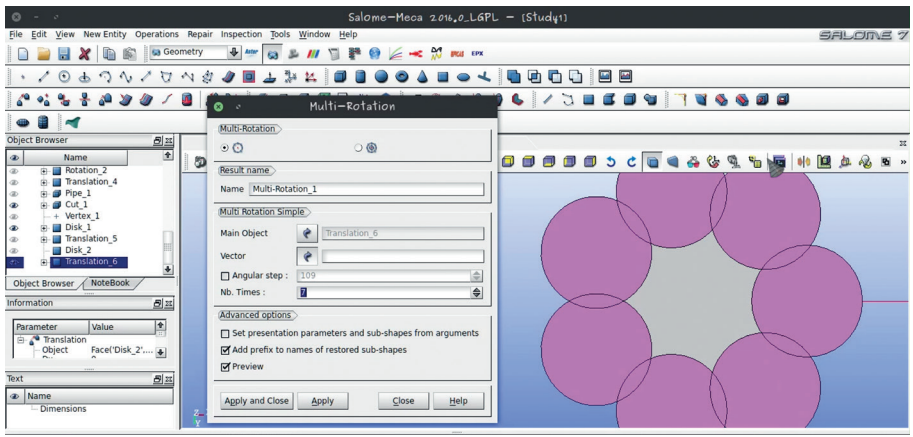


Figura 150. *Multiplicar disco 1*

Enseguida accedemos a New Entity – Explode. Esta herramienta será vital para visualizar los vértices de la misma. Figura 150.

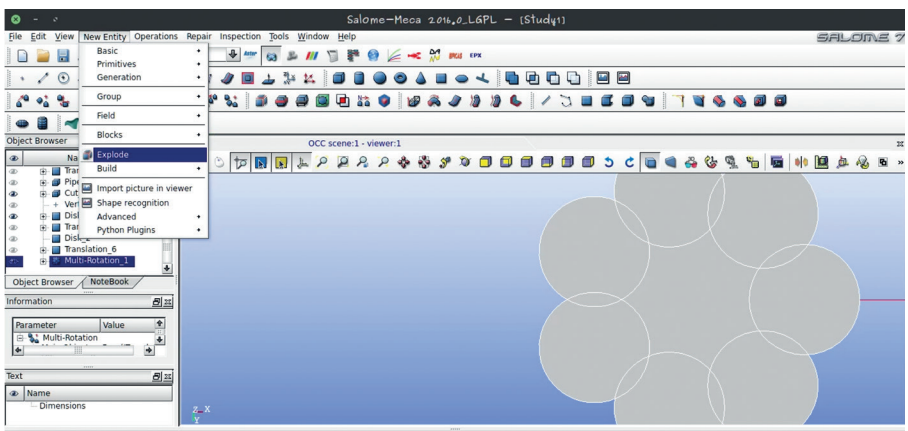


Figura 151. *Visualizar vértices*

En su ventana Sub-shapes Selection nos vamos a los argumentos en Main Object elegiremos Vertex, aplicamos y cerramos. Figura 152.

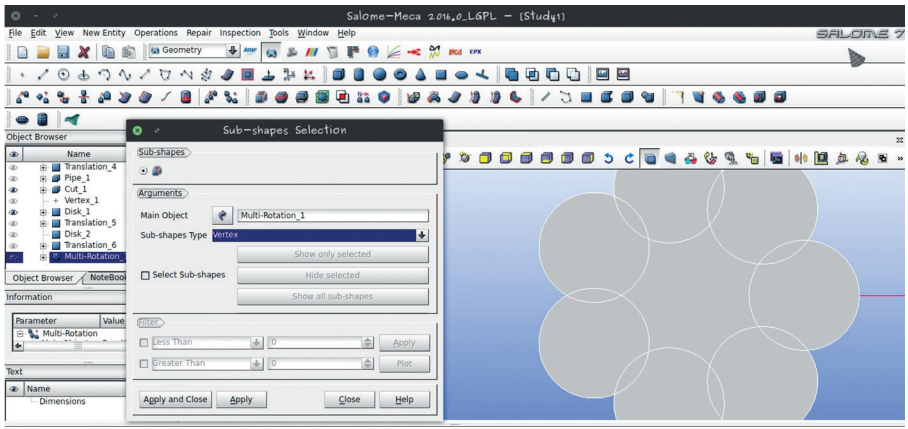


Figura 152. *Aplicar Vertex*

El resultado de la operación anterior mostrada, enseguida, solo presenta los vértices.

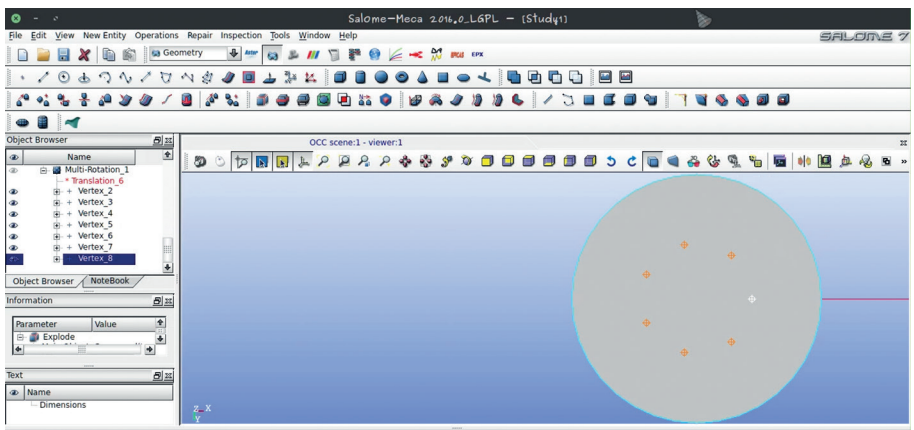


Figura 153. *Visualizar vértices*

Con la ayuda de la herramienta Line (Línea) uniremos cada uno de los vértices en sentido de las manecillas del reloj para formar nuestro heptágono, como se manifiesta en la figura siguiente. Al finalizar cerramos nuestra ventana. Figura 154.

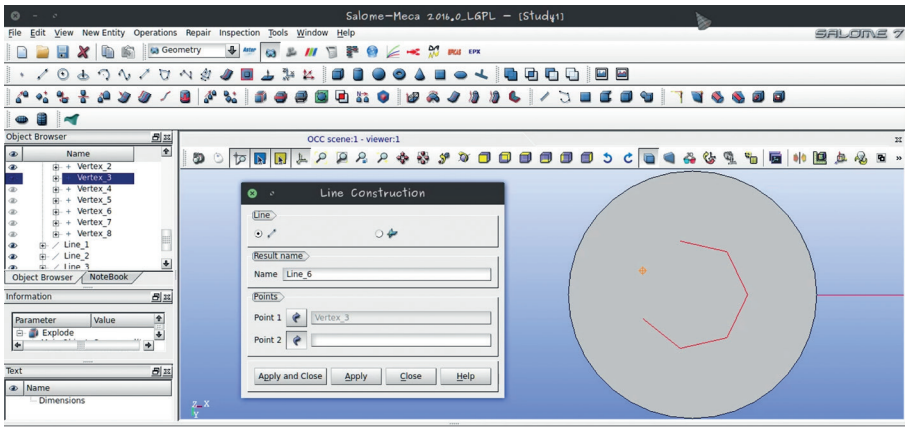


Figura 154. *Unir los vértices*

Una vez creado nuestro heptágono lo que sigue es fusionar estas líneas para crear un solo elemento, para ello nos dirigimos al menú Operations – Boolean – Fuse. Figura 155.

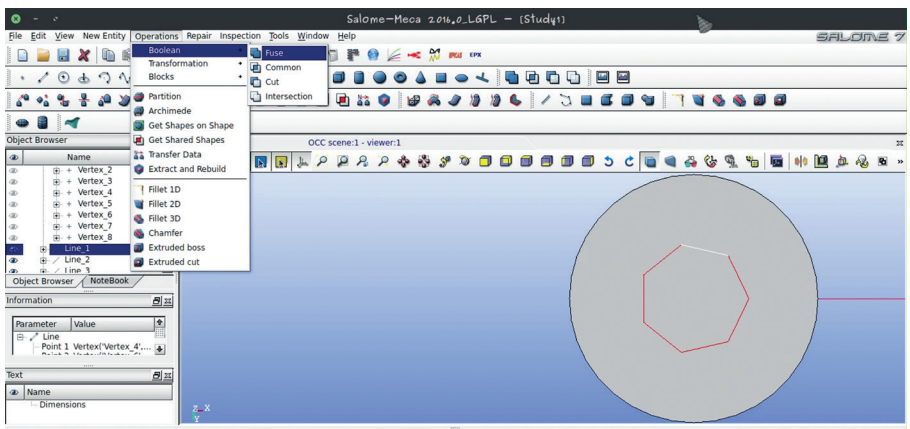


Figura 155. *Crear un solo elemento*

Podemos observar en la ventana Fuse Object que en la parte de sus argumentos seleccionamos todos los objetos que deseamos fusionar, en este caso las siete líneas creadas para nuestro heptágono, aplicamos y cerramos. Figura 156.

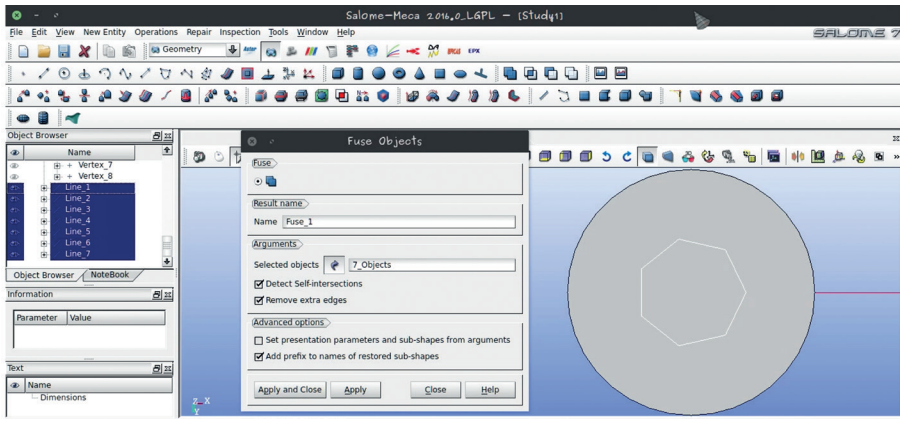


Figura 156. *Selección de objetos fusionados*

Ahora tenemos un heptágono, pero como se puede observar está muy pequeño para poder ser la cabeza de nuestro tornillo, por lo cual tendremos que escalar nuestro heptágono, entonces nos vamos al menú Operations – Transformation – Scale Transform.

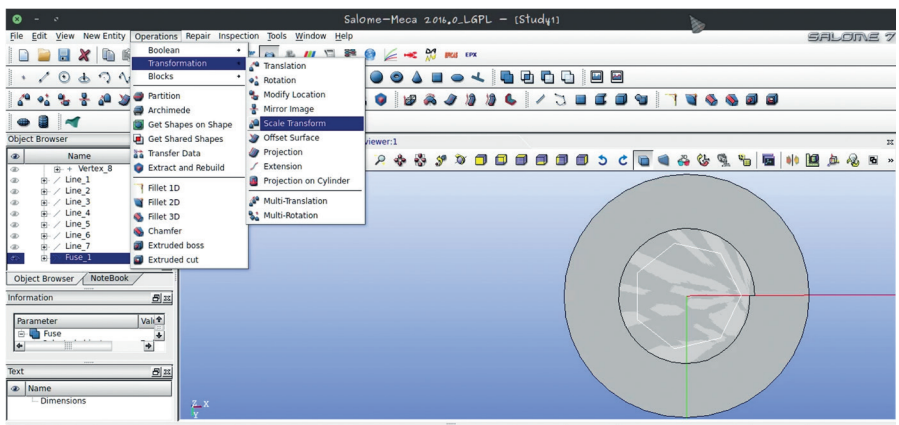


Figura 157. *Resultado del heptágono*

Usando la opción primera en nuestra ventana Scale an Object podemos escalar nuestro objeto y después recurrimos a los argumentos. Como primer paso seleccionamos el objeto a escalar (Objects), y en la opción Scale Factor podemos elegir el tamaño a escalar; en este caso lo haremos a 2.5. Como se

puede observar en la figura siguiente, un trapecio con el contorno rosa aparece sobre el extremo de nuestro disco grande, indicando para este caso en específico el tamaño deseado. Aplicamos y cerramos. Figura 158.

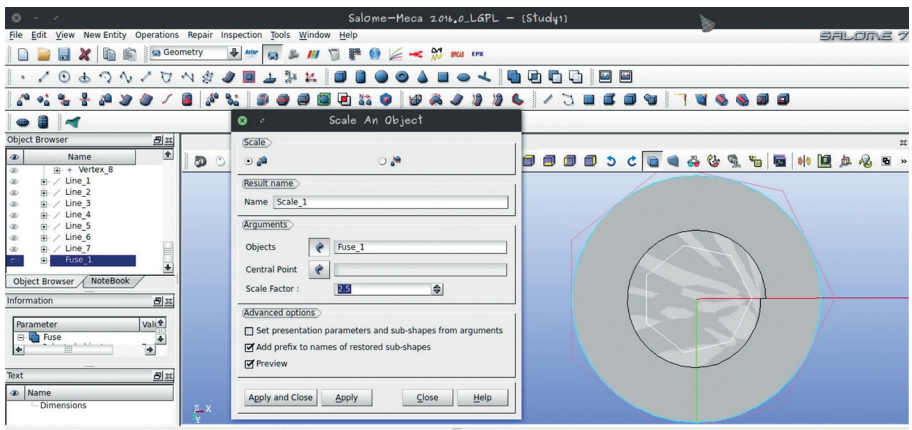


Figura 158. *Escalar objeto*

El paso siguiente es darle una cara a nuestro heptágono, para ello nuevamente usaremos la herramienta Create a Face, elegimos el elemento, aplicamos y cerramos. Figura 159.

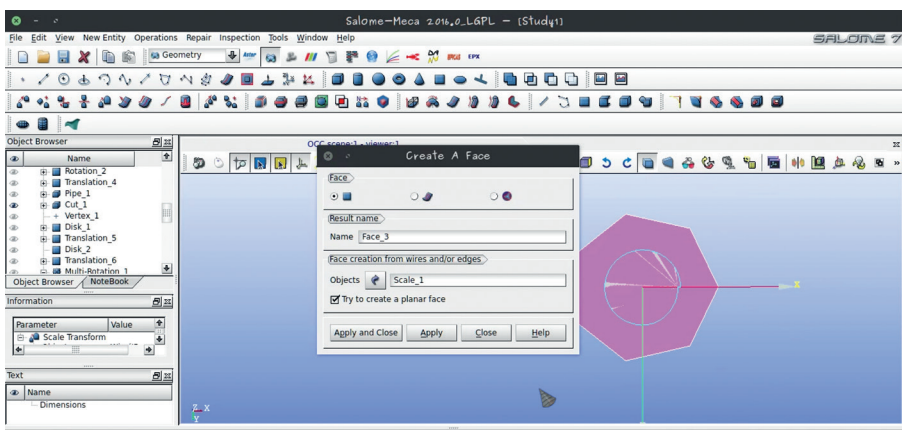


Figura 159. *Crear una cara al heptágono*

A continuación, para crear una extrusión nos dirigimos al menú New Entity – Generation – Extrusion desde donde podremos apreciar su funcionamiento en la figura 160.

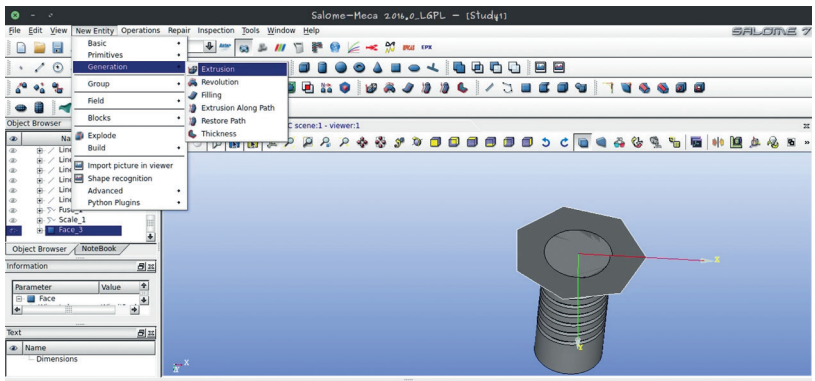


Figura 160. *Extrusión*

Al crear la extrusión podemos observar la ventana Construction by Extrusion. Como primer punto elegimos la tercera opción. Sobre Base Shape + DX DY DZ Vector encontramos algunas opciones, elegimos nuestro objeto a extrusionar, más abajo encontramos las diferentes direcciones DX, DY y DZ, modificaremos las opciones de estos hacia donde deseemos extrudir; para este ejercicio lo extruiremos sobre DZ, pero será hacia abajo, por lo que utilizaremos números negativos, aplicamos y cerramos, como se ilustra en la figura 161.

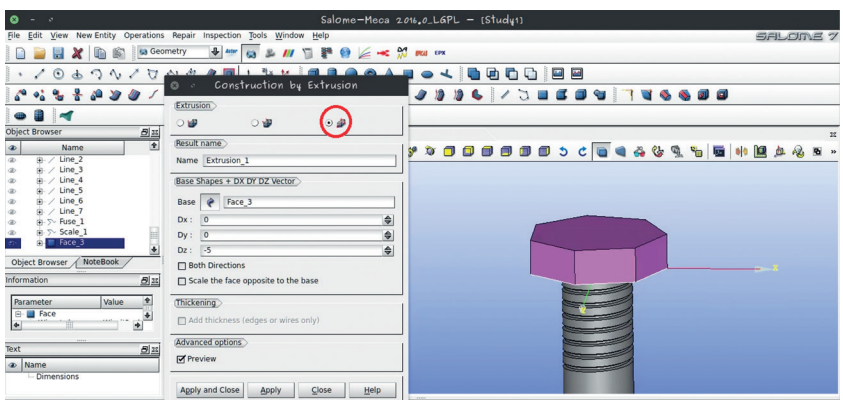


Figura 161. *Base Shape + DX DY DZ*

Como se puede observar, la cabeza de nuestro tornillo se encuentra en la parte de abajo. Necesitamos colocarlo en la parte de arriba, para ello haremos uso de la herramienta Translation como anteriormente lo hemos visto. En la ventana Translation Of An Object elegiremos la tercera opción. Sobre los argumentos tenemos Objects, seleccionamos el elemento que deseamos mover, sobre Vector elegimos el vector que creamos anteriormente, aplicamos y cerramos. Figura 162.

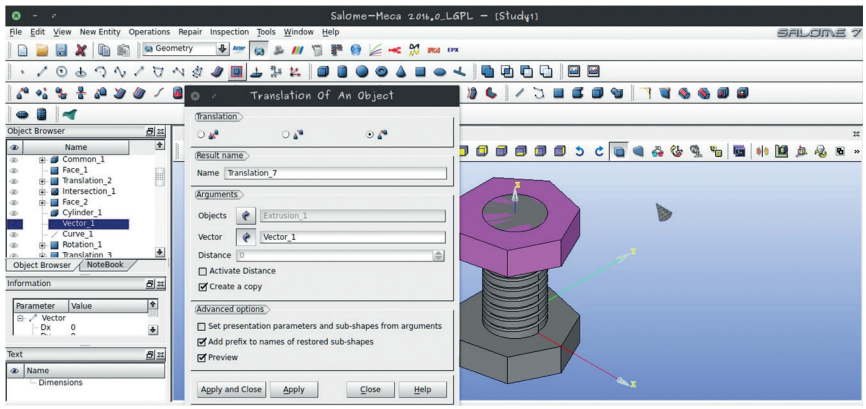


Figura 162. *Cabeza del tornillo*

Al llegar a este punto, podemos observar que nuestro tornillo está casi terminado. Figura 163.

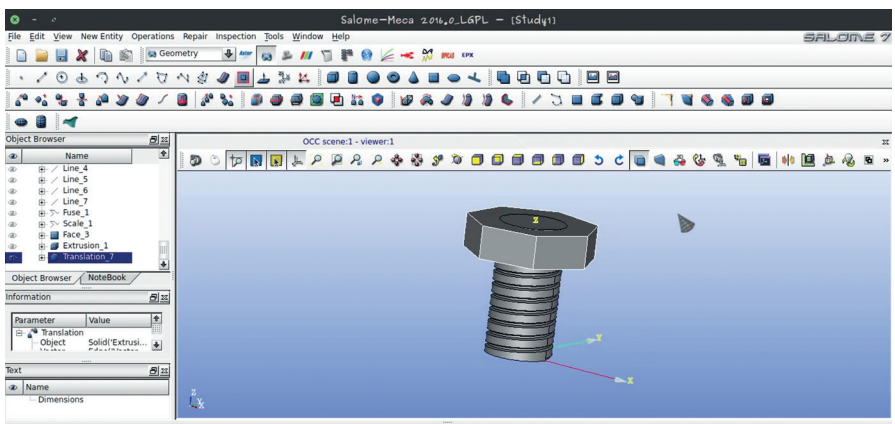


Figura 163. *Tornillo casi terminado*

Detalles del tornillo

Cuando llegamos a este punto quedan por realizar los pequeños detalles, para ello nos dirigimos a Operations – Chamfer y continuamos.

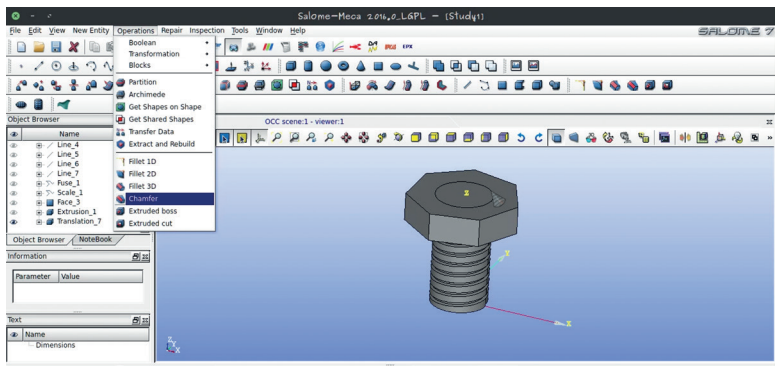


Figura 164. *Detalles del tornillo*

En la ventana Chamfer Construction elegimos la tercera opción, sobre Chamfer On Selected Faces encontramos el Main Object en el cual elegimos el objeto para la operación, en la parte Selected Faces elegimos las caras de la figura; si observamos el tornillo en la figura de abajo, tenemos las caras de su cabeza seleccionadas. En el contorno ya se encuentra lo que nos indica que estas piezas están seleccionadas y elegimos el diámetro, finalizamos y cerramos. Figura 165.

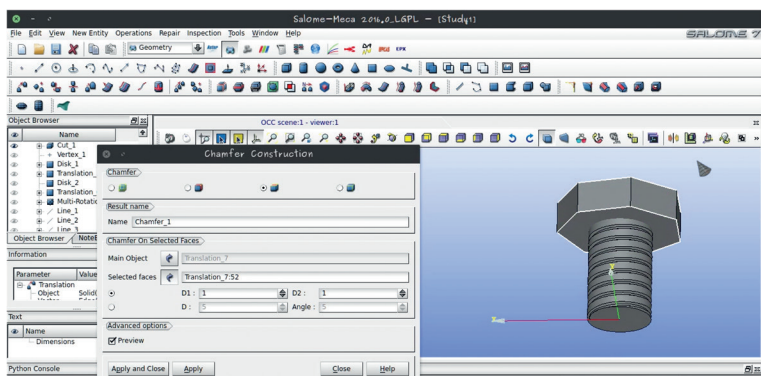


Figura 165. *Chamfer Construction*

El resultado lo veremos en la figura 166.

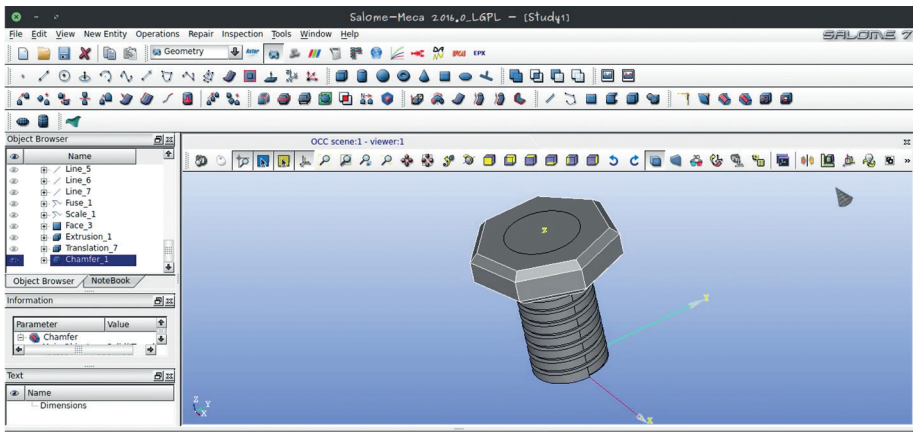


Figura 166. *Tornillo casi terminado*

Como paso final fusionaremos las piezas como anteriormente lo hemos hecho. Figura 167. Hasta aquí los ejemplos de la conformación de geometrías diversas, a partir de las primeras herramientas descritas del programa Salome Meca. Corresponde, entonces, continuar con la enunciación de otros mecanismos sustanciales, como el mallado, a partir del último ejemplo geométrico del tornillo.

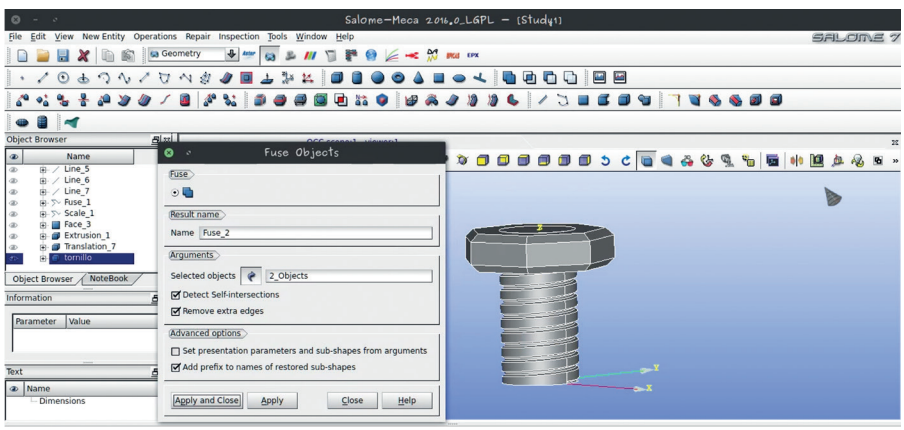


Figura 167. *Tornillo fusionado y terminado*

Malla (Mesh)

El módulo Mesh contiene un conjunto de algoritmos de mallado que se utilizan para las entidades de mallado (1D, 2D, 3D sub-formas) que componen objetos geométricos. Mesh representa una aproximación discreta de un subconjunto del espacio de tres dimensiones por geometrías elementales.

Un estudio puede contener múltiples mallas y cada una de ellas puede ser utilizada (exportada) de forma individual.

Proporciona varias maneras de crear una malla. La forma principal es la construcción de la malla sobre la base de la forma geométrica que se produce en el módulo de la geometría. De esta manera:

- un objeto geométrico (forma principal);
- parámetros de mallado (algoritmo de mallado y características (por ejemplo, el tamaño del elemento) de una malla requerida, encapsulado en hipótesis de objetos).

Las sub-mallas permiten discretizar algunas sub-formas de la forma principal, por ejemplo una cara, usando los parámetros de mallado que difieren de las utilizadas para otras sub-formas, pueden ser editadas usando las funciones destinadas a la modificación.

La malla puede ser importada o exportada a partir de formatos MED, VNU, STL, CGCs, DAT, GMF, etcétera.

La estructura de una malla se describe por los nodos y elementos. La geometría de un elemento se define por la secuencia de nodos que lo constituyen y de la convención de conectividad.

La malla puede incluir las siguientes entidades:

- **Nodo** es una entidad de red que define una posición en el espacio 3D con coordenadas (X, Y, Z).
- **Edge** (o segmento) es un elemento de malla 1D que une dos nodos.
- **Cara** es un elemento de malla 2D que representa una parte de la superficie unida por enlaces entre nodos de la cara. Una cara puede ser un triángulo, un cuadrángulo o un polígono.
- **Volumen** es un elemento de malla 3D que representa una parte del espacio 3D obligado por las facetas de volumen. Un volumen puede ser un tetraedro, hexaedro, pentaedro, pirámide, prisma hexagonal o poliedro.
- **0D** es el elemento de malla definida por un nodo.

SALOME soporta elementos de segundo orden, sin nodo central (triángulo cuadrático, cuadrángulo, polígono, tetraedro, hexaedro, pentaedro y pirámide) y con nodos centrales (triángulo, cuadrángulo y hexaedro).

Iniciar módulo Mesh

Es necesario elegir un algoritmo de mallado para cada dimensión de sub-formas hasta la dimensión más alta que se generará.

Antes de continuar, al tener lista nuestra geometría podemos acceder al módulo Mesh de 2 formas en la figura 167 encerradas en un cuadro rojo, ahí se encuentran las formas de acceder al módulo.

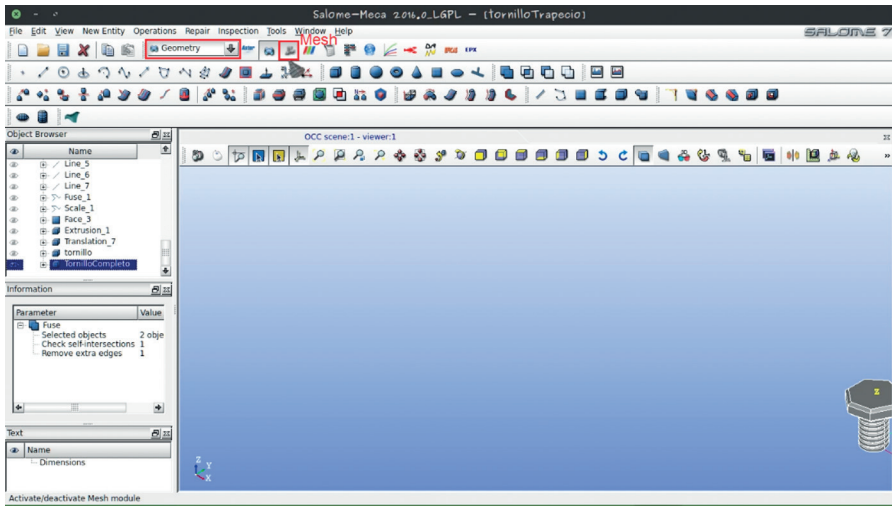


Figura 168. *Formas para acceder al módulo*

Una vez que accedamos, veremos que su interfaz es parecida al módulo Geometry antes vista con la gran diferencia de que sus herramientas son distintas, iremos analizando el uso básico de este módulo. Figura 169.

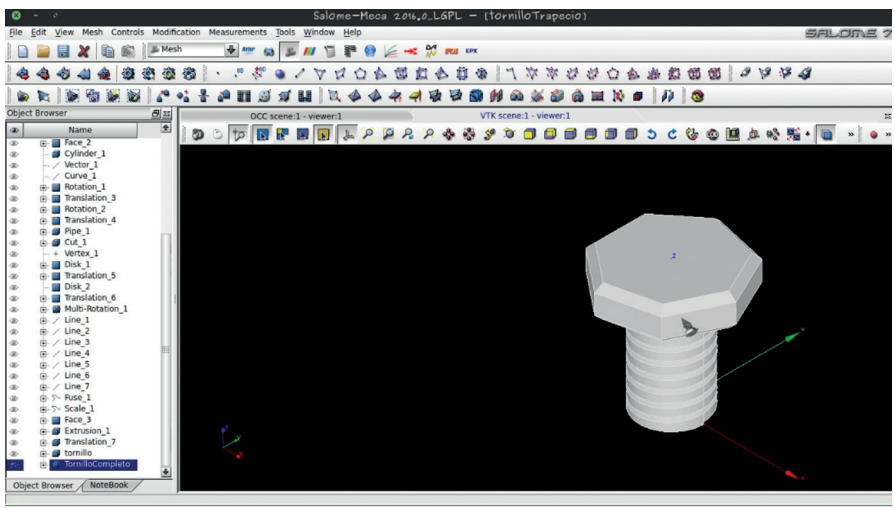


Figura 169. *Interfaz al módulo Geometry*

Generación de malla

La generación de malla en la geometría se realiza en el flujo ascendente: primero se crean nodos en vértices, luego las aristas se dividen en segmentos usando nodos en vértices; el nodo de segmentos se ocupa entonces para acoplar caras; entonces los nodos de las caras son necesarios para engranar sólidos. Esto asegura automáticamente la conformidad de la malla.

Podemos crear nuestra malla en el ícono  (Create mesh) ubicado en la barra de herramientas; de igual forma, podemos ir al menú Mesh > Create Mesh.

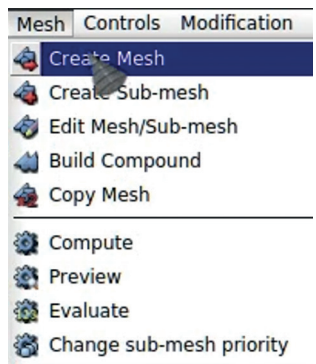


Figura 170. *Ícono Crear malla*

Ahí nos mostrará una ventana Create mesh, como primer argumento (Name) escribiremos un nombre o dejaremos el que se maneja por defecto.

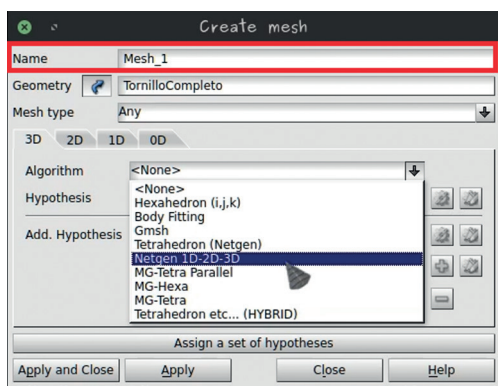


Figura 171. *Nombrar al argumento*

El segundo argumento (Geometry) es elegir la geometría que requerimos, le damos clic sobre la flecha azul y después elegimos el elemento en el explorador de elementos (ventana Object Browser) figura 172.

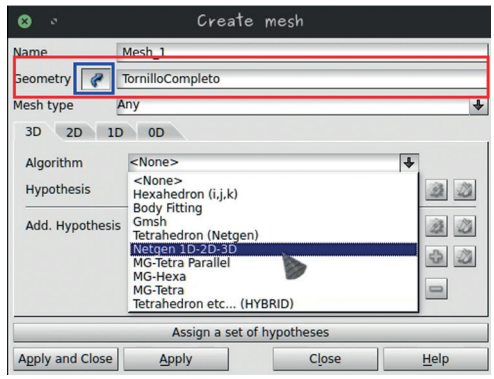


Figura 172. *Elegir la geometría a utilizar*

El tercer argumento (Mesh type) se puede elegir desde alguna opción dando clic en la flecha encerrada en el círculo azul, hay que tener en cuenta que la configuración la determinará el sistema o podemos configurarla manualmente. Figura 172.

Entre las opciones 0D, 1D, 2D y 3D, elegimos 3D.

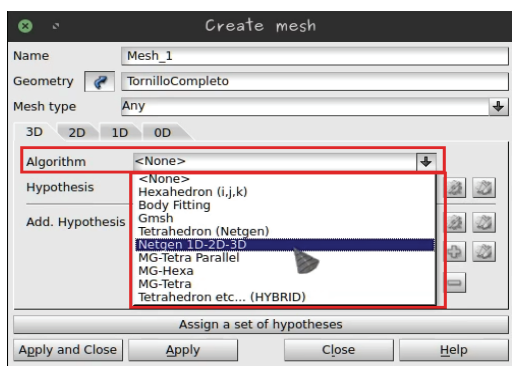


Figura 173. *Seleccionar un argumento*

Para el cuarto argumento (Algorithm) correspondiente a nuestra figura tridimensional, elegiremos la opción Netgen 1D-2D-3D.

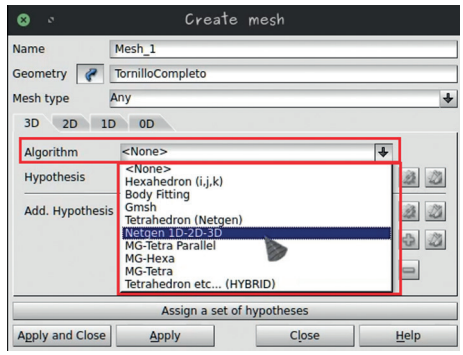


Figura 174. *Seleccionar algoritmo*

En el quinto argumento (Assign a set of hypothesis) damos clic sobre este y elegimos la opción 3D: Automatic Tetrahedralization y continuamos. Figura 175.

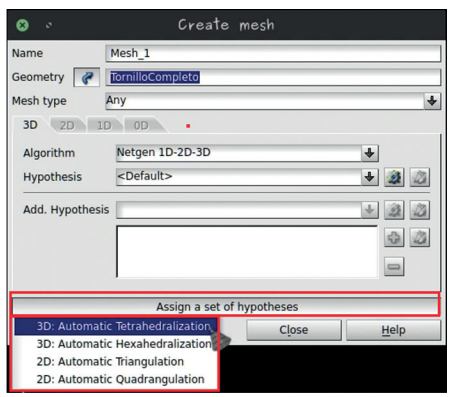


Figura 175. *Assign a set of hypothesis*

Nos saldrá una ventana nueva Hypothesis Construction, la renombramos por defecto. Aparacerá el argumento Length al que le podemos indicar un valor, debemos tener en cuenta que si el valor es muy chico, nuestro cálculo tendrá mayor número de elementos calculados, en cambio si nuestro valor es mayor, nuestra malla tendrá menos cálculos. Una vez elegido, clic en Ok; aplicamos y cerramos.

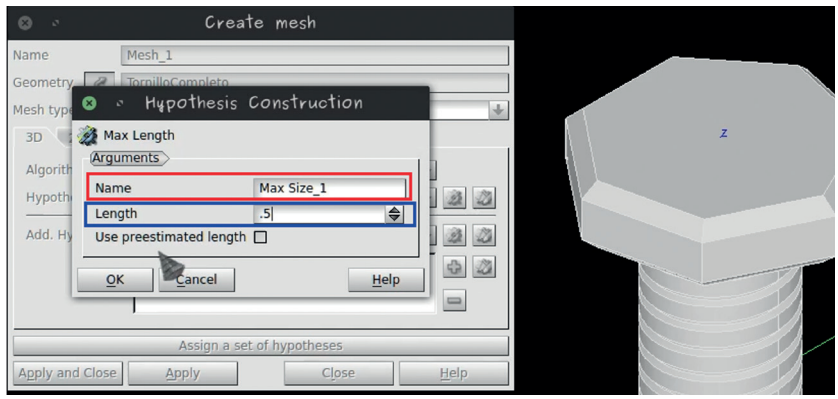


Figura 176. *Hypothesis Construction (length)*

Después de crear el objeto malla haremos el cálculo de la malla con un clic en el ícono compute (calcular)  ubicado en la barra de herramientas y esperamos el procesamiento de dato.

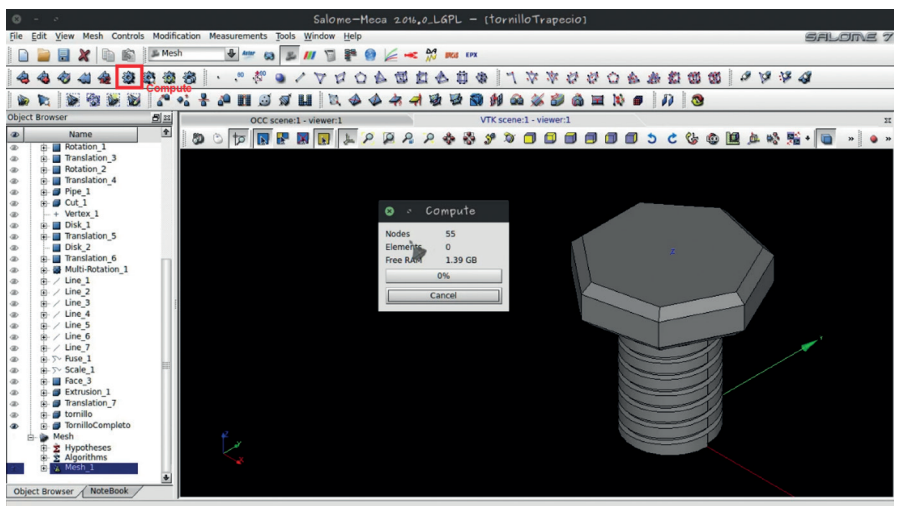


Figura 177. *Cálculo de la malla*

Una vez finalizado el cálculo de malla, aparece el cálculo en la ventana Mesh computation succeed. Si cierra esta casilla y vuelve a hacer clic en el botón Calcular, sin modificar previamente los parámetros de mallado, la malla

NO se volverá a calcular y se mostrará el cuadro Información de malla con el mismo contenido.
Si el cálculo de malla ha sido un éxito, en la ventana Mesh computation succeed nos mostrará la información sobre el número de entidades de diferentes tipos en la malla. Figura 177.

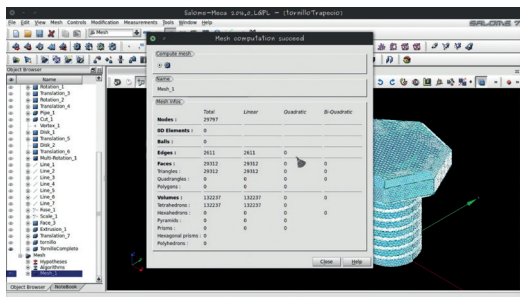


Figura 178. *Número de entidades*

Exportación de malla

El sistema cuenta con una funcionalidad que permite la importación y exportación de mallas MED, UNV (I-DEAS 10), DAT (formato ASCII simple), STL, GMF (formato interno de los productos DISTENE, archivos de formato CGNCS Y SAUV.

Para llevar a cabo la exportación debemos seleccionar el elemento a exportar, ir al menú File (Archivo) elegimos Export (exportar) y optamos por cualquier formato; en nuestro interés, usamos el formato MED.

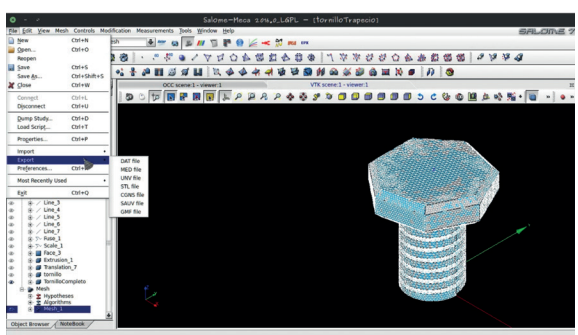


Figura 179. *Formato MED*

En la búsqueda de archivos estándar se debe seleccionar una ubicación para el archivo a exportar y escriba su nombre. Guardamos dando clic en Save.

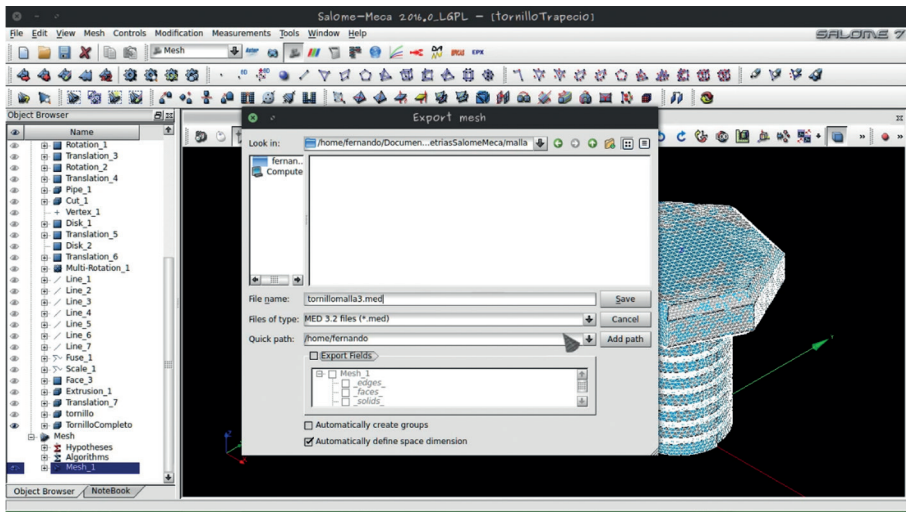


Figura 180. *Seleccionar ubicación*

Existen parámetros adicionales disponibles en la exportación a archivos de formato MED y SAUV.

- **Creación automática de grupos** (Automatically create groups) especifica si se deben crear grupos de todas las entidades de malla, de dimensiones o no disponibles. Si está marcada, los grupos creados tienen nombres como Group_On_All_Nodes o Group_On_All_Faces.
- **Definir de forma automática la dimensión espacial** (Automatically define space dimension) especifica si se debe definir la dimensión espacial para la exportación por la configuración de malla o no. Por lo general, la malla se exporta como una malla en el espacio 3D, al igual que en el módulo de malla.

Acabamos de ver la forma de crear un elemento malla, configurarla y realizar el cálculo de discretización del elemento y exportación. A continuación veremos algunas herramientas y utilidades más de este módulo.

Diferencia de longitud (Length) en la malla

Crearemos una nueva malla usando la misma geometría (TornilloCompleto) y editamos Assign a set of hypothesis, elegimos la opción 3D: Automatic Tetrahedralization; en ella modificamos el valor de Length = 10, clic en Ok, aplicamos y cerramos.

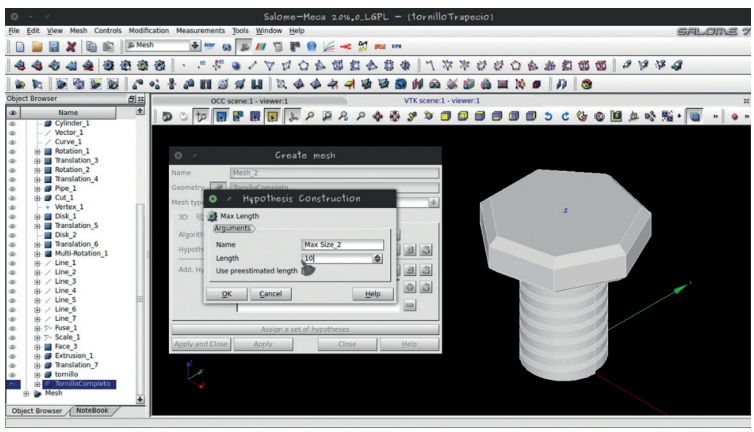


Figura 181. *Crear nueva malla*

Ahora realizamos el cálculo con clic en Compute y esperamos a que termine. Figura 182.

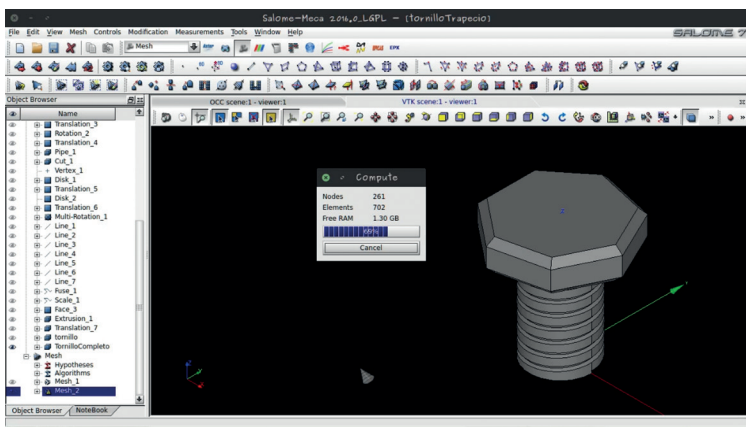


Figura 182. *Realizar cálculo*

Si observamos los resultados, en la longitud de 10 podemos crear una menor cantidad total de 2,413 tetraedros, más abajo comparamos los resultados entre una longitud mayor y una menor.

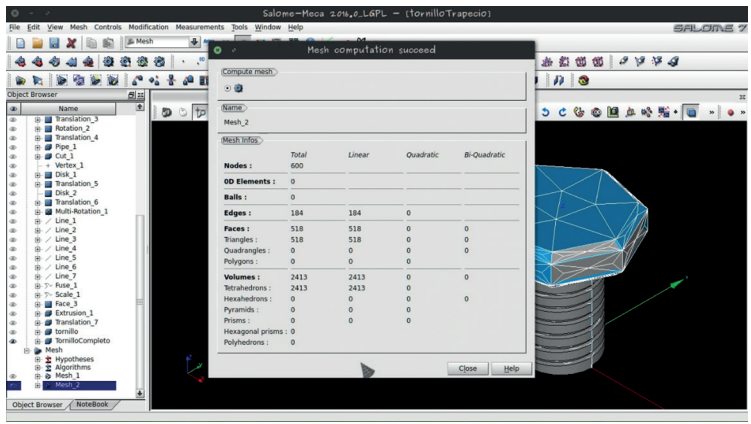


Figura 183. *Resultados obtenidos*

Aquí podemos observar la comparación entre los valores de longitudes como anteriormente fue mencionado entre la longitud más chica. El cálculo será mayor, como se muestra en la figura 182: hemos creado una malla de longitud .5, la cual desarrolló un total de 132,237 tetraedros; en cambio, en la figura 183 hemos creado una malla de longitud 10, la cual ha calculado un total de 2,413 tetraedros; la configuración de estos argumentos se hará de acuerdo con las necesidades de nuestros proyectos.

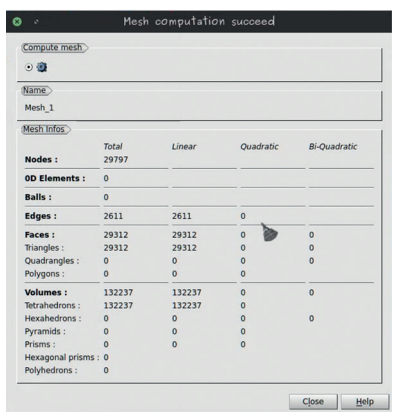


Figura 184. *Longitud de .5*

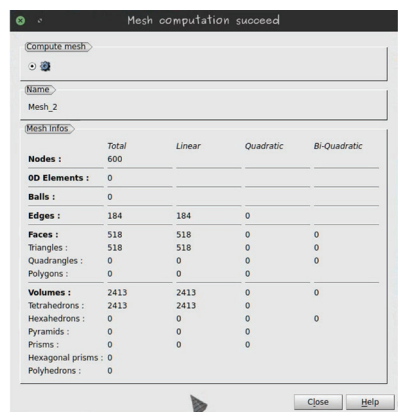


Figura 185. *Longitud de 10*

Conjuntos de hipótesis (Assign a set of hypothesis)

Ahora describiremos tres conjuntos de hipótesis: Triangulation, Hexahedralization y Quadrangulation.

Triangulation

Creamos una malla a nuestro tornillo completo y en el argumento Assign a set of hypothesis elegimos la opción 2D: Automatic Triangulation.

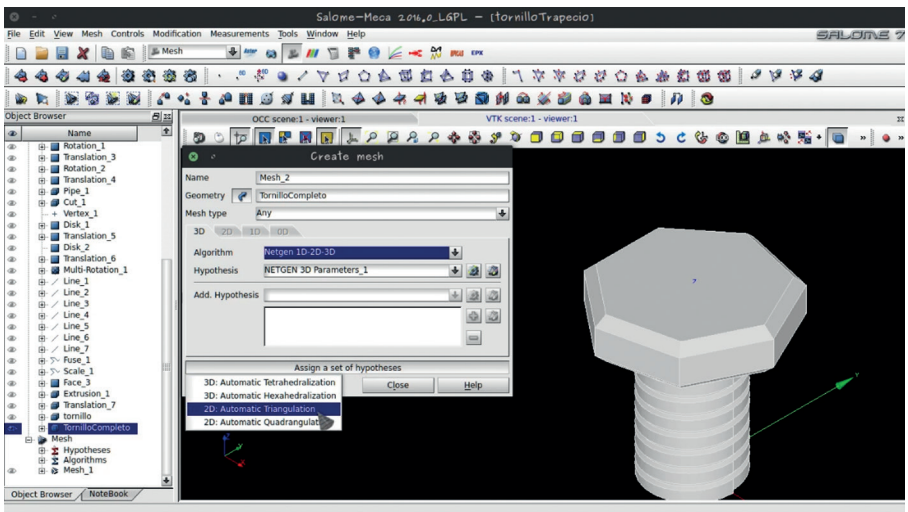


Figura 186. *Aplicar la triangulación*

Renombramos o usamos por defecto la longitud (Length) que calculamos en .5, damos clic en Ok, aplicamos y cerramos. Posteriormente llevamos a cabo el cálculo con la herramienta Compute. Figura 187.

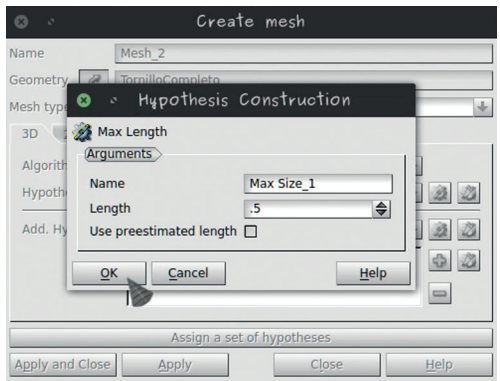


Figura 187. *Renombrar y cambiar longitud*

Observando el cuadro de información Mesh computation succeed podremos apreciar que se han creado 29312 de forma triangular y las demás entidades. Figura 188.

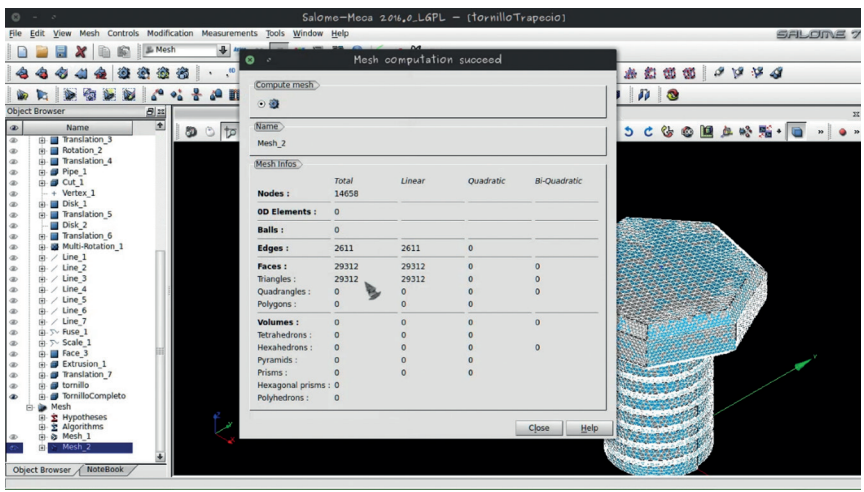


Figura 188. *Visualizar entidades*

Hexahedralization

Creamos una malla a nuestro tornillo completo y en el argumento Assign a set of hypothesis, elegimos la opción 3D: Automatic Hexahedralization.

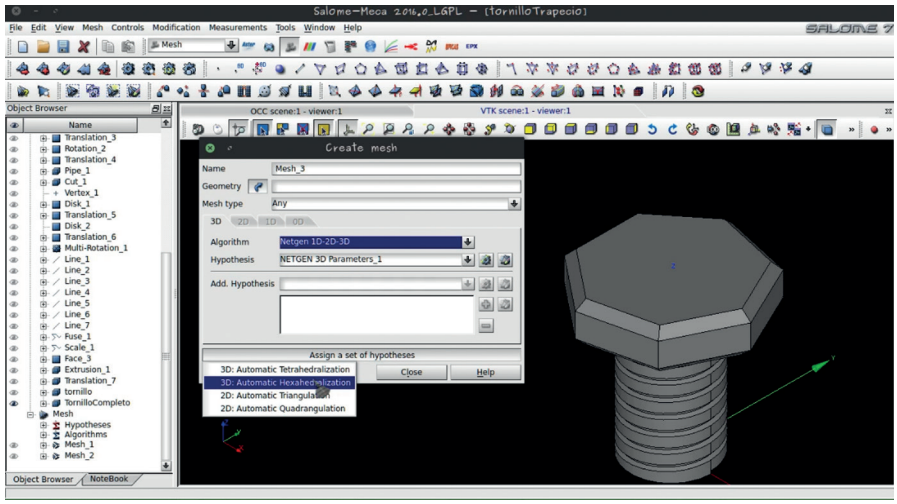


Figura 189. *Malla al tornillon Hexahedralization*

En esta parte lo vamos a mantener y damos clic en Ok, después aplicamos y cerramos, posteriormente procedemos a computar la malla. Figura 189.

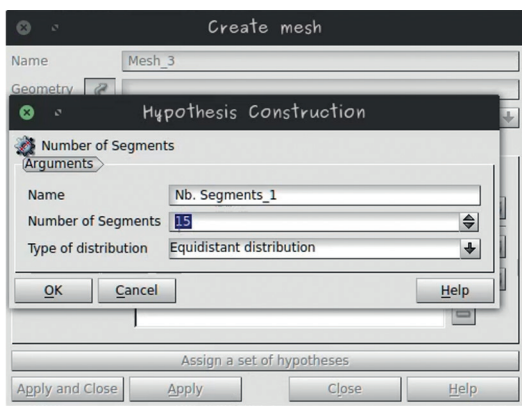


Figura 190. *Mantener los datos iguales*

Podemos observar los resultados que nos proporciona este tipo de hipótesis en la figura 190.

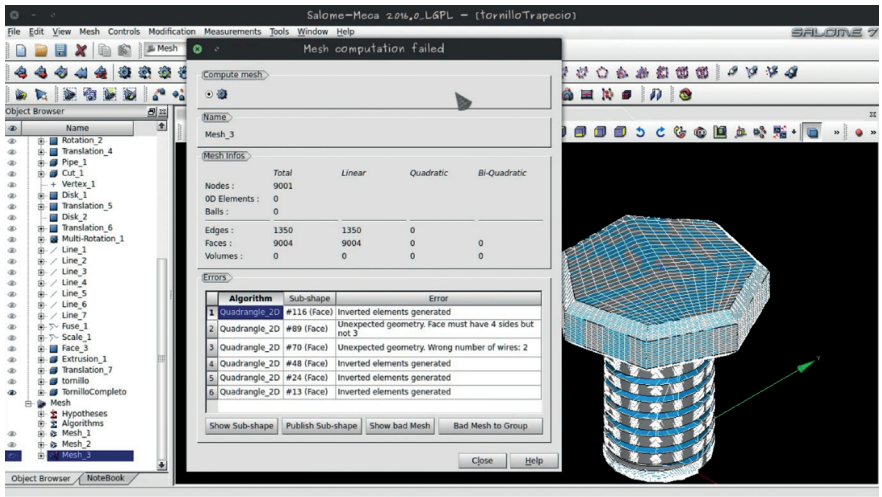


Figura 191. *Resultados de la hipótesis*

Quadrangulation

Creamos una malla a nuestro TornilloCompleto y en el argumento Assign a set of hypothesis, elegimos la opción 2D: Automatic Quadrangulation. Figura 192.

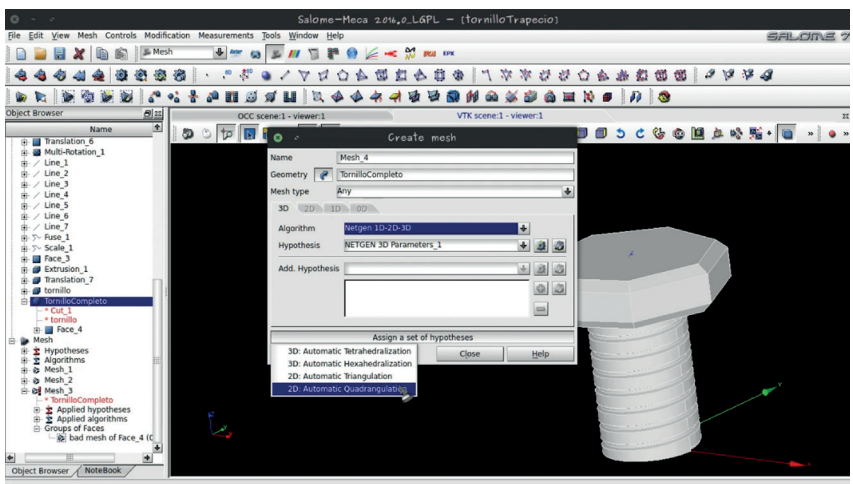


Figura 192. *Crear malla en el tornillo Quadrangulation*

Sobre los argumentos que usaremos, debemos tener en cuenta que podemos modificarlos según nuestros requisitos.

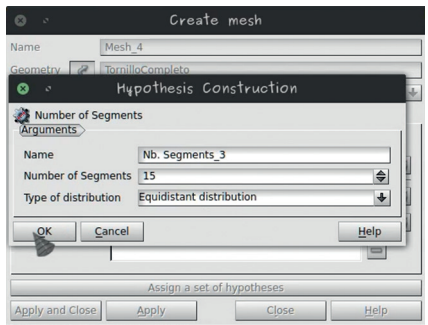


Figura 193. *Argumentos a utilizar*

Una vez establecidos nuestros argumentos podemos llevar a cabo el cálculo y observar los resultados.

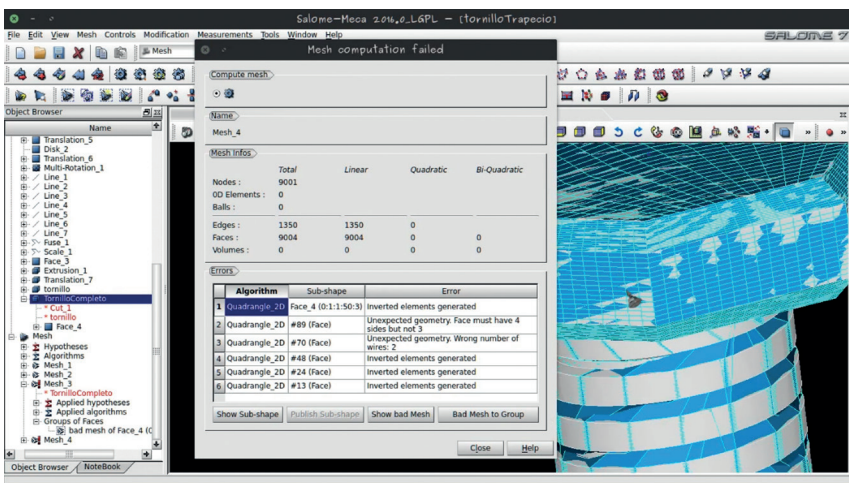


Figura 194. *Visualizar cálculo*

Construcción y configuración de la hipótesis

Haga clic en el botón  que corresponde a añadir la hipótesis para configurar sus partes. Elegimos Netgen 3D Parameters. Figura 195.

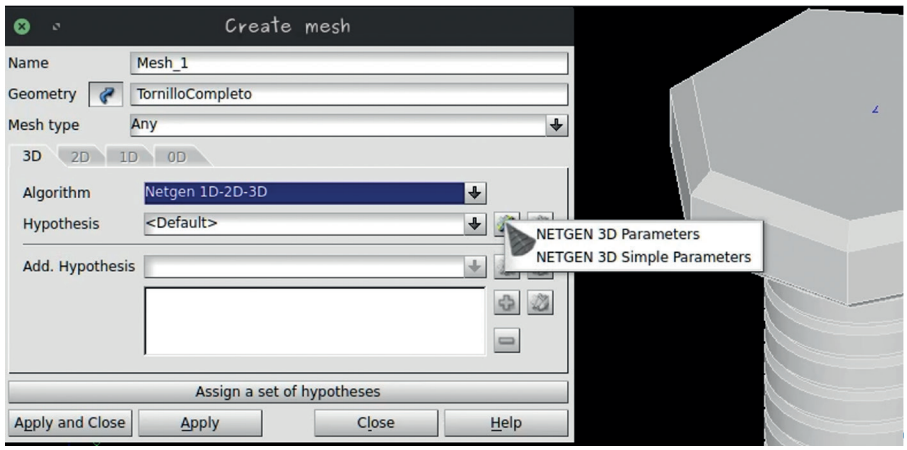


Figura 195. *Añadir hipótesis*

Sobre esta ventana Hypothesis Construction tenemos dos apartados; sólo analizaremos brevemente los Arguments: Name, elegimos el nombre de la hipótesis a construir. Max. Sise y Min. Sise son los tamaños mínimos y máximos. Sobre Fineness podemos ver varias opciones desde un mallado muy grueso a muy fino, de igual forma puede ser personalizado. En este caso elegimos Very Coarse. Cliqueamos en Ok, aplicamos y cerramos. Figura 196.

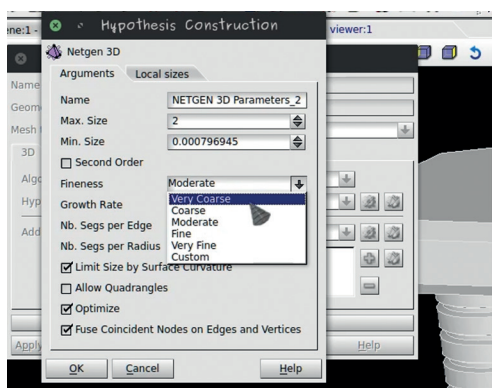


Figura 196. *Análisis de argumentos*

Observemos los resultados en la figura 197, nos creó 24,430 tetraedros, el resultado es muy diferente a los derivados anteriormente.

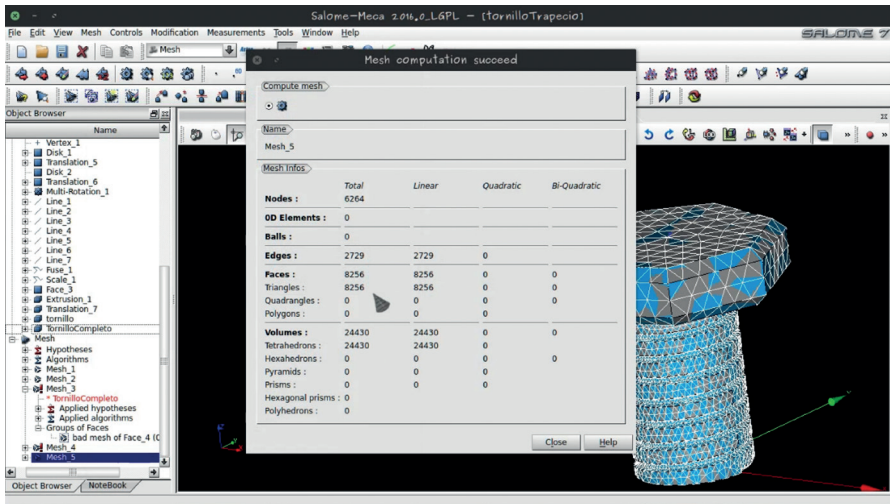


Figura 197 *Resultados obtenidos*

Editar una Mesh

Para llevar a cabo la edición de una malla, damos clic en el ícono "Edit Mesh" que se encuentra en la barra de herramientas o sobre el menú > Mesh > Edit Mesh/Sub-Mesh o dando clic derecho sobre la malla modificar y elegir la opción Edit Mesh/Sub-Mesh.

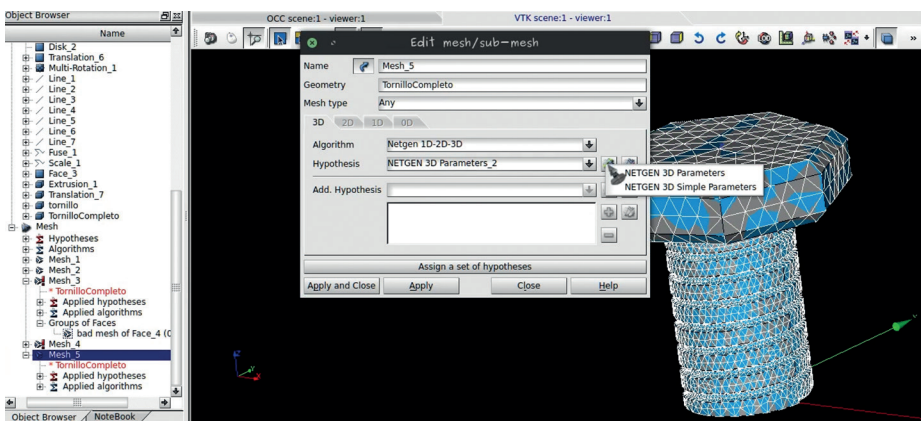


Figura 198. *Editar una malla*

Como podemos observar, nuestro objeto está con una malla que hemos creado anteriormente, enseguida elegimos sobre Netgen 3D Parameters.

Los argumentos máximos son de dos y el mínimo es por defecto sobre Fineness, seleccionamos la opción Very Fine y damos Clic en Ok, en aplicar y cerrar. Posteriormente llevamos a cabo el cálculo. Figura 199.

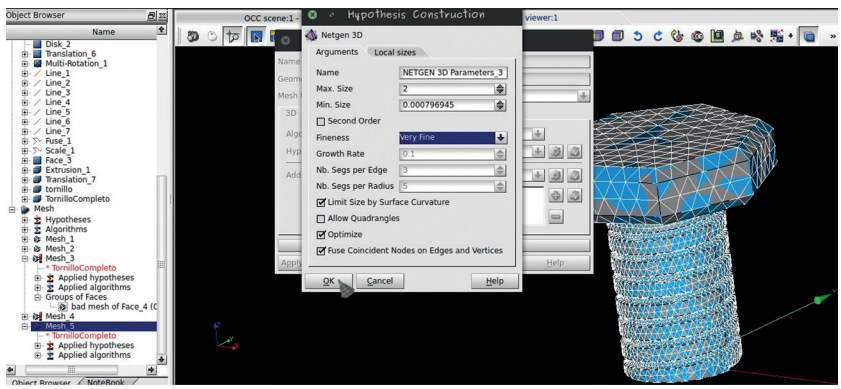


Figura 199. *Very Fine*

Si observamos los resultados, nos ha creado un número de tetraedros mayor que el primer estudio realizado con una construcción de hipótesis (Assign a set of hypothesis) extendida en una longitud de .5, calculando 132,237 tetraedros. Con una malla fina ha creado 278,874 tetraedros.

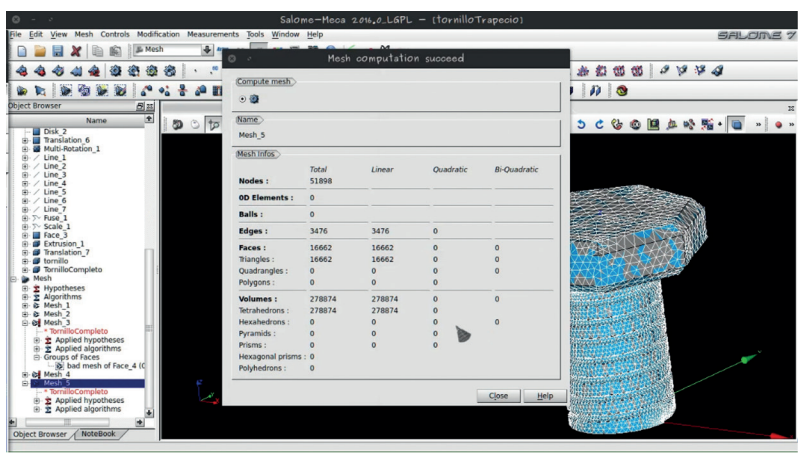


Figura 200. *Resultado en número de tetraedros*

Posteriormente creamos una malla nueva haciendo uso de la segunda opción que nos ofrece la configuración de hipótesis NETGEN 3D Simple Parameters, como se muestra en la figura siguiente.

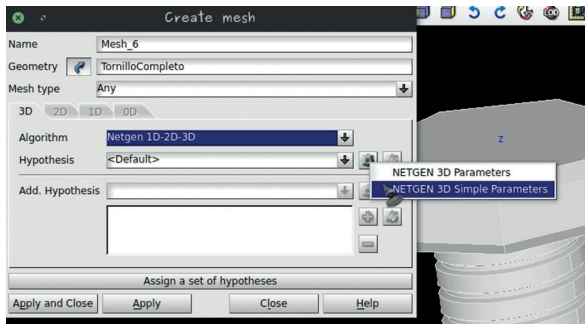


Figura 201. *Malla nueva con la segunda opción*

Esta opción nos muestra la ventana Hypothesis Construction, en la cual sólo analizaremos el argumento Number of Segments y lo mantendremos en 30; reiteramos los demás argumentos y aceptamos. Posteriormente hacemos el cálculo.

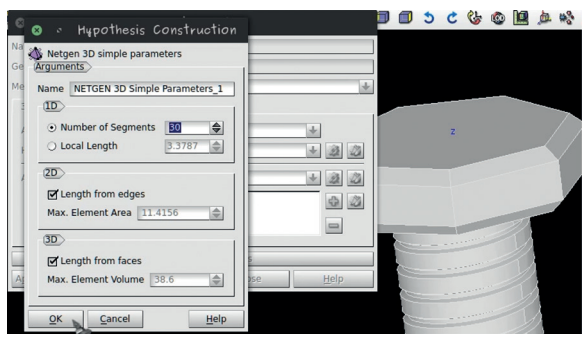


Figura 202. *Argumento con números en seguimiento*

Así queda nuestro resultado con parámetros simples:

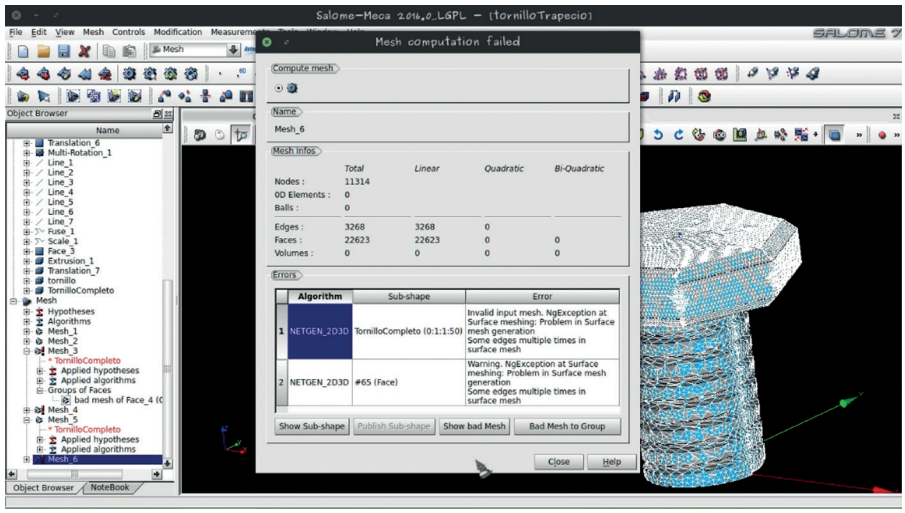


Figura 203. *Resultados en parámetros simples*

Errores

Si el cálculo de la malla ha fallado, la información sobre la causa de la falla se proporciona en la tabla Errores. De la misma forma que se muestra en el recuadro rojo sobre la figura 203.

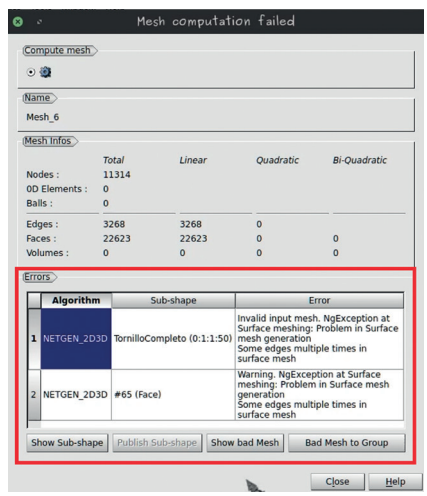


Figura 204. *Muestra de error*

La categoría Mostrar Sub-forma (Show sub-shape) nos permite visualizar en magenta el enhebrado de la entidad geométrica que falló (el nombre de esta entidad o su ID y tipo de muestra en la columna Sub-forma).

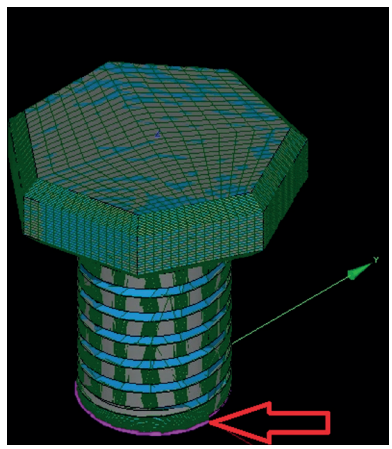


Figura 205. *Visualización del magenta*

La malla incorrecta (Bad Mesh to Group) muestra las entidades de malla que se vieron entorpecidas desde el visor en magenta.

Con publicar Sub-forma (Publish Sub-shape) se podrá identificar el mallado que ha fallado en la geometría, lo que permite analizar la geometría problemática y crear una sub-malla para ajustar localmente las hipótesis.

Hasta aquí dejamos la práctica de geometrías y damos paso a la instalación de Code Saturne con su consecuente ejemplificación de la simulación de una lámpara de neón. Hemos querido dar a conocer una parte esencial de objetos simulados que corresponden a la geometría de los artefactos, hemos también transitado por la malla que se añade para el uso del método de elementos finitos. Ahora consideraremos otras variables que influyen en el diseño de prototipos, como la temperatura, el gas, la energía, la presión o la entalpía.

Instalación de Code Saturne 2.0.4

Para realizar una buena instalación del programa Code Saturne, es recomendable utilizar el sistema operativo Caelinux GNU/Linux http://caelinux.com/CMS/, así mismo la versión 2.0.4 de Code Saturne.

Los archivos necesarios se encuentran en dos carpetas llamadas installer e installer11, que serán instaladas en el sistema operativo Caelinux. Figura 205.

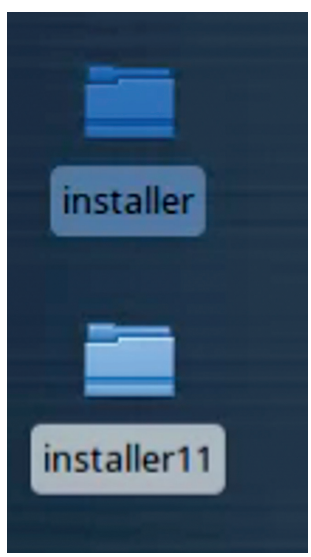


Figura 206. *Archivos a utilizar*

En nuestra carpeta installer encontraremos tres archivos como se muestra en la figura 207.

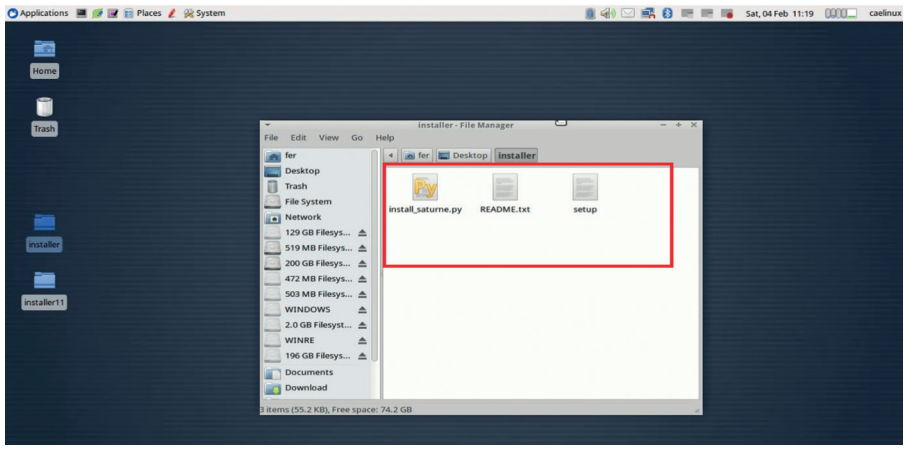


Figura 207. *Tres archivos a encontrar*

Abriremos nuestra terminal en el directorio installer y ejecutaremos el archivo install_saturne.py, con el comando ./install_saturne.py según se observa en la figura 208.

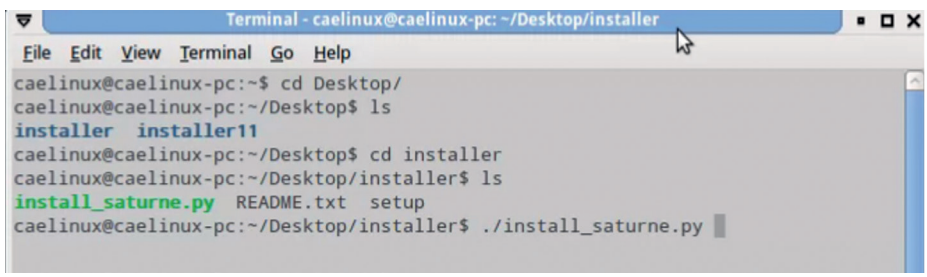


Figura 208. *Terminal en el directorio*

Nos mostrará un error encerrado dentro de un recuadro en rojo, el cual nos indica que nos falta el directorio opt que será creado más adelante. Figura 209.

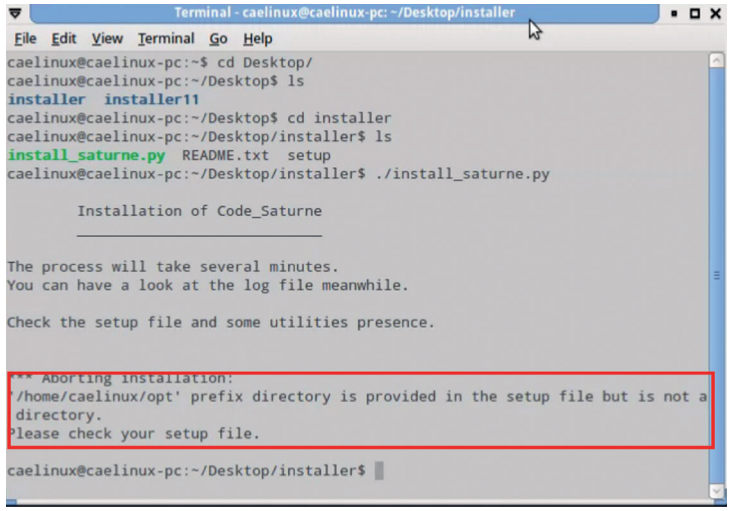


Figura 209. *Error mostrado*

Crearemos el directorio opt como se muestra en la figura 210.

Para continuar con la instalación dentro del directorio /installer editaremos el archivo setup con el editor Leafpad mostrado enseguida.

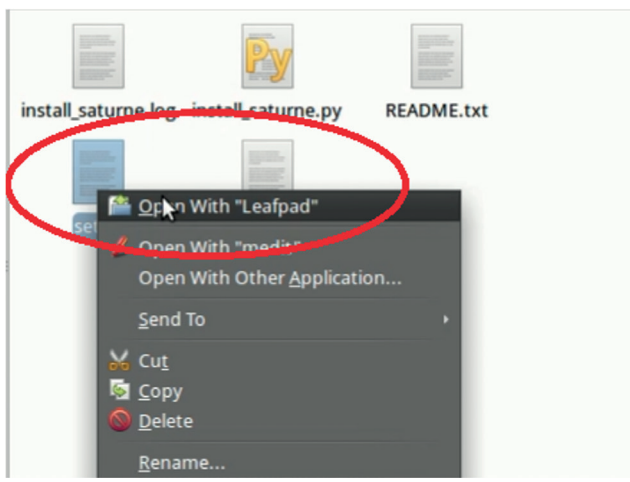


Figura 210. *Directorio opt*

En esta parte cambiaríamos la forma de instalación de descarga. Cambiaremos de download yes a download no. Figura 211.

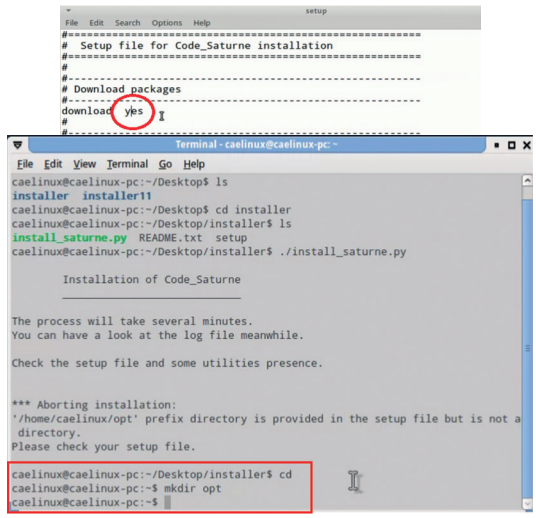


Figura 211. *Download no*

El siguiente paso será copiar los archivos que se encuentran en el directorio installer11 a installer. Figura 212.

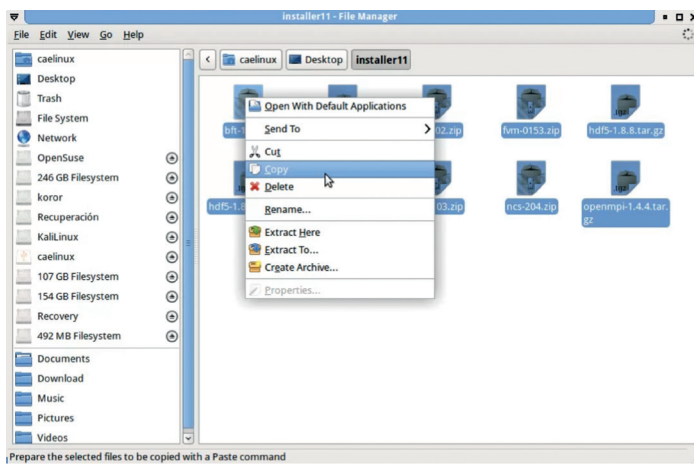


Figura 212. *Copia de archivos*

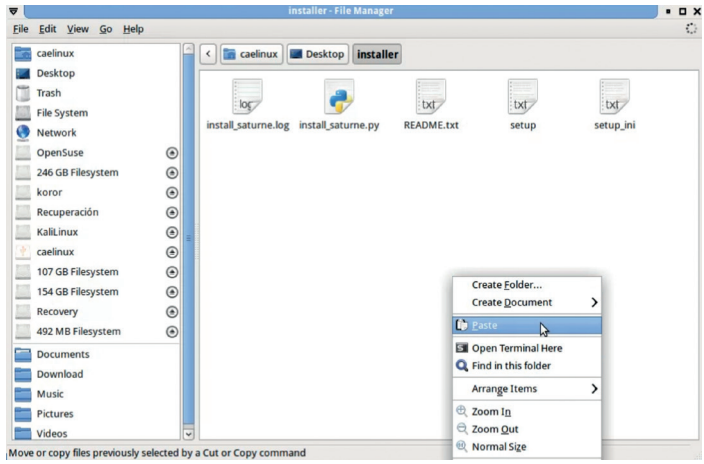


Figura 213. *Pegar archivos*

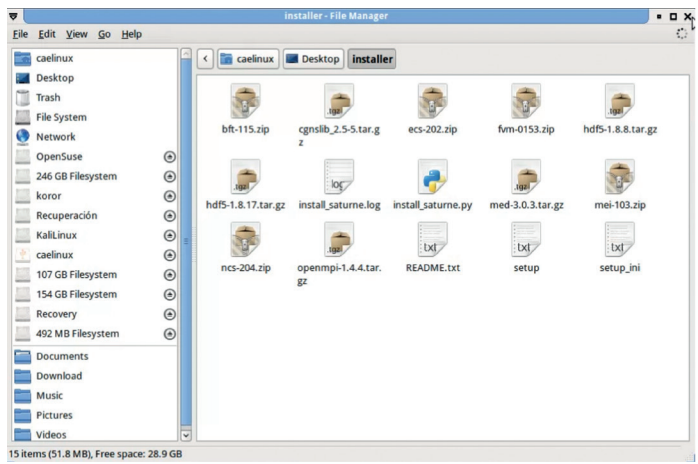


Figura 214. *Archivos copiados*

Si ejecutamos de nuevo el archivo install_saturne.py encontraremos un problema más, cuya solución encontraremos más adelante. Figura 215.

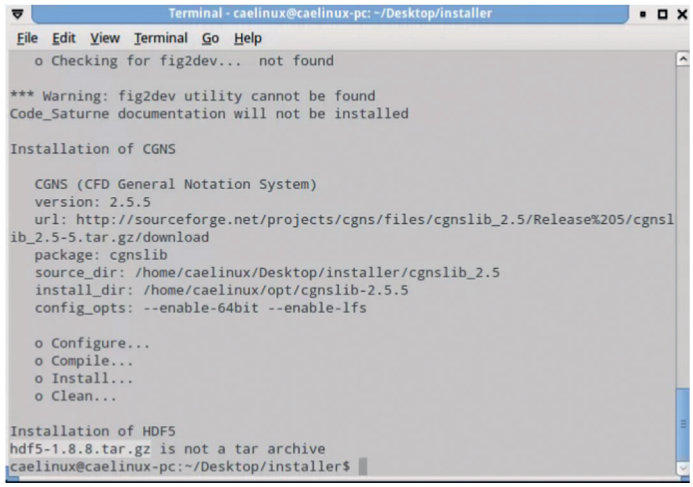


Figura 215. *Problema mostrado*

Para solucionar este problema editaremos con Leafpad (cualquier editor que sea de su agrado), los archivos setup e install_saturne.py, en donde cambiaremos la versión del hdf5-1.8.8 a hdf5-1.8.17, según se muestra en la figura 216.

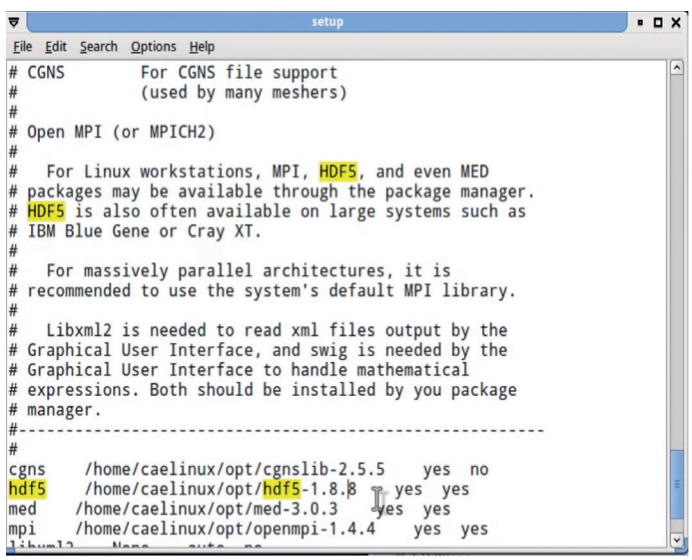


Figura 216. *Solución del problema*

Esos son los ajustes y configuraciones que se necesitan para realizar la instalación de nuestro programa. A continuación, ejecutaremos nuevamente nuestro archivo./install_saturne.py en nuestra terminal, como se muestra en la figura 217.

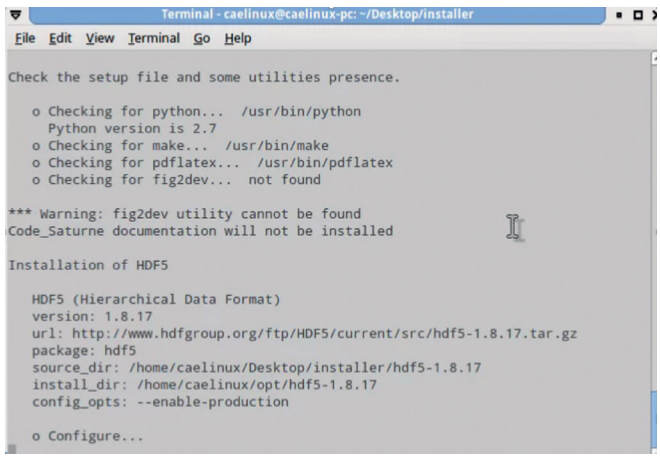


Figura 217. *Ejecución*

En la figura 218 aparece enmarcado en un recuadro azul el mensaje de que la instalación ha terminado. El proceso de instalación se demora dependiendo de cada equipo de cómputo.

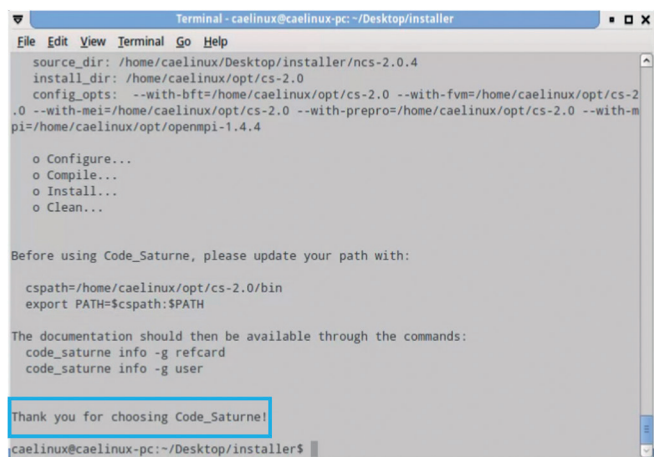


Figura 218. *Mensaje de instalación terminada*

Para el buen funcionamiento de Code Saturne con la terminal, hay que configurar el path; para ello copiaremos dos líneas de texto que aparecen encerradas en el recuadro verde de la figura 218. Posteriormente hay que pegarlas.

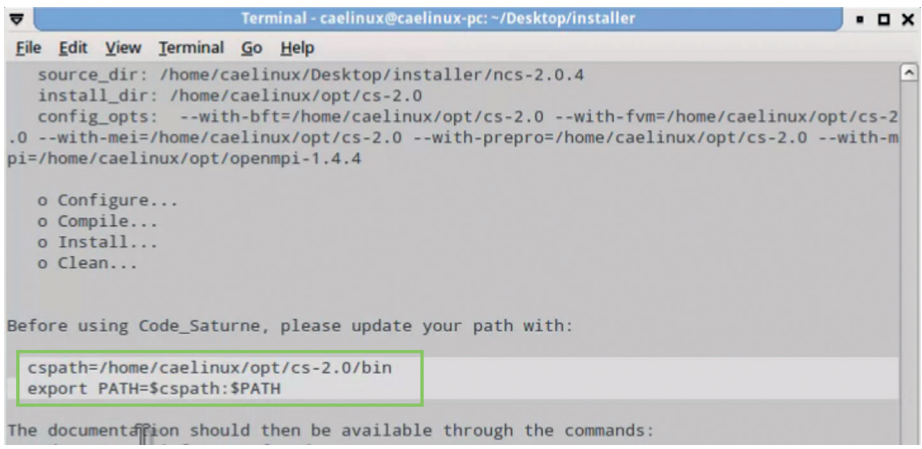


Figura 219. *Configuracion de Path*

Se pegarán en el archivo .bachrc oculto en el directorio principal. Para visualizar presionaremos ctrl+h y editamos el archivo con el editor preferido. Figura 219.

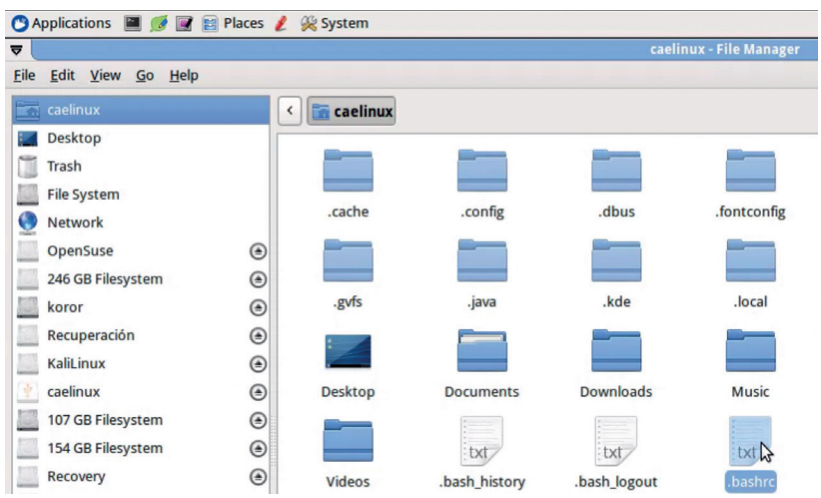


Figura 220. *Archivo bachrc*

A continuación, escribiremos nuevamente esas líneas en nuestra terminal como se muestra en la figura 220 y Code Saturne estará listo para usarse.

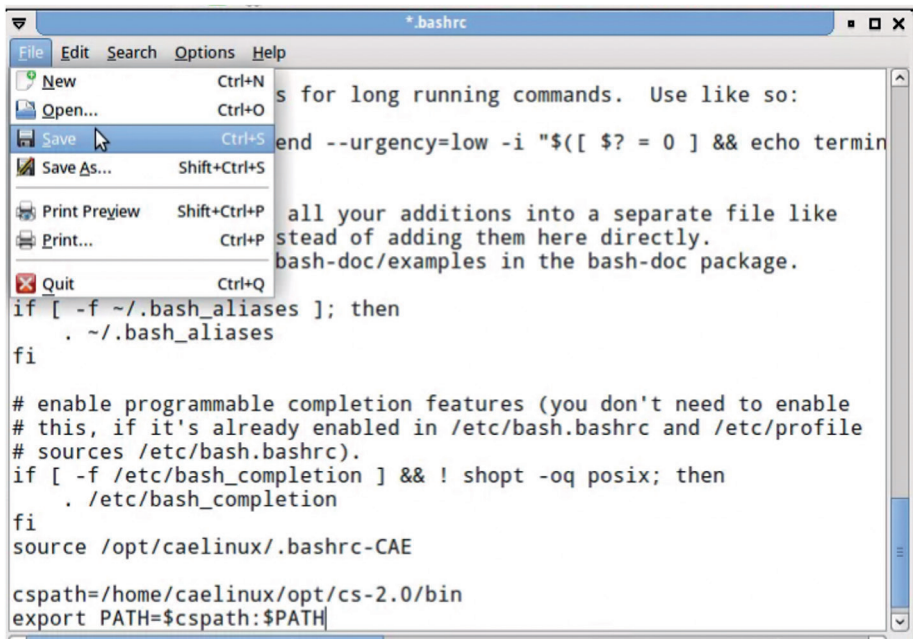


Figura 221. *Escribir en las terminales*

Simulación de un plasma frío para lámpara de neón

Creación de la geometría

Para llevar a cabo la simulación de un plasma frío necesitaremos una geometría que realizaremos con ayuda de Salome Meca; para ello creamos un cilindro con las dimensiones siguientes: el radio será de 10 cm y el largo de 100 cm. Figura 222.

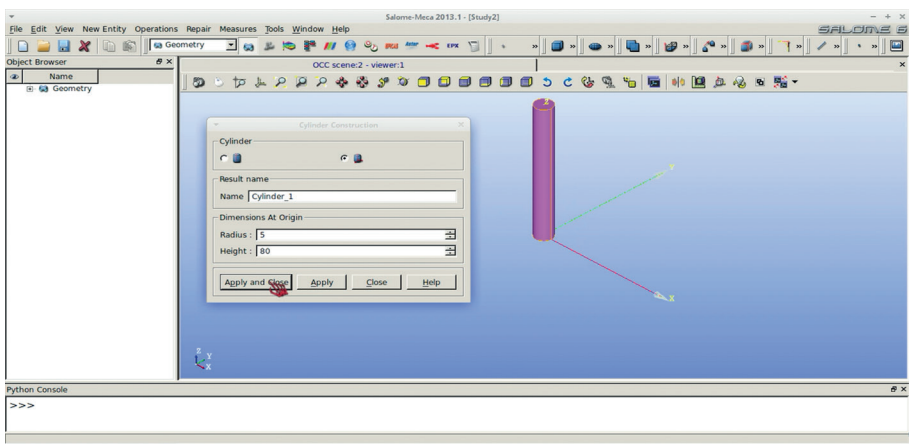


Figura 222. *Creación de la geometría*

Aplicación de malla

Una vez concluida nuestra figura le aplicaremos una malla como se muestra en la figura 223.

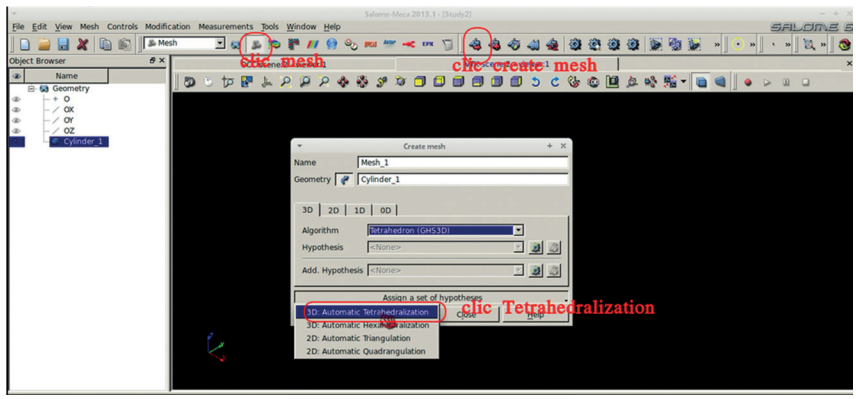


Figura 223. *Aplicación de malla*

Deberemos elegir la longitud más pequeña posible para un mayor número de tetraedros, ya que de esta forma obtendremos un estudio más detallado. Si observamos la figura 224, veremos que nuestro mallado contiene 21,981 tetraedros.

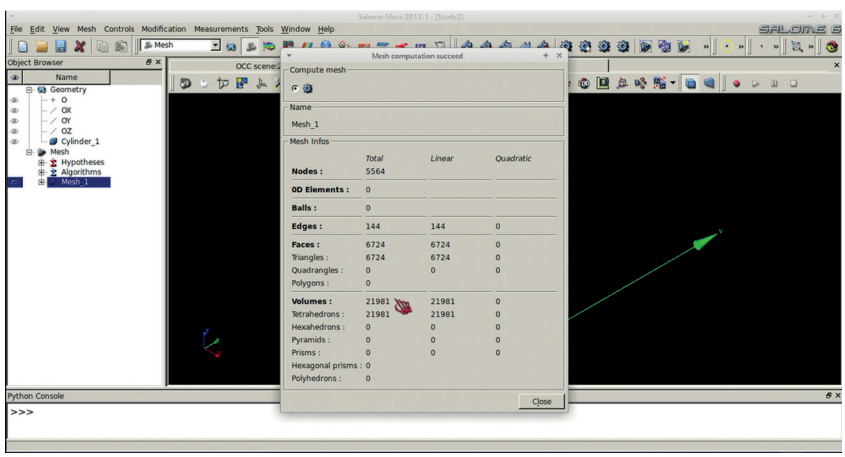


Figura 224. *Longitud pequeña*

Una vez concluido el número de tetraedros podremos exportar nuestra malla a la carpeta MESH de nuestro estudio, el cual lo explicaremos más adelante.

Para ello nuestro estudio deberá estar ya creado. Figura 225.

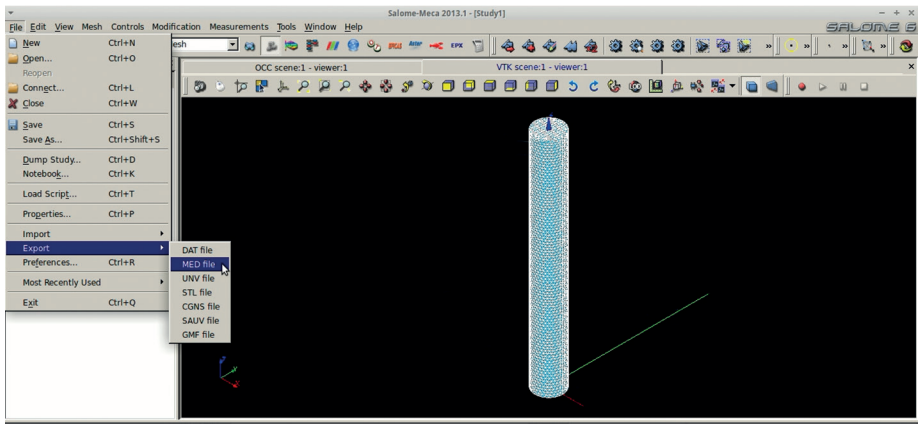


Figura 225. *Estudio ya creado*

Creación del estudio con Code Saturne

Comenzaremos abriendo una terminal; usaremos la ubicación por defecto, a continuación, ingresamos el siguiente comando code_saturne create lampara --nogui, en la cual (la palabra lampara es el nombre del estudio, puede ser cualquier nombre), podemos observarlo en la figura 226.

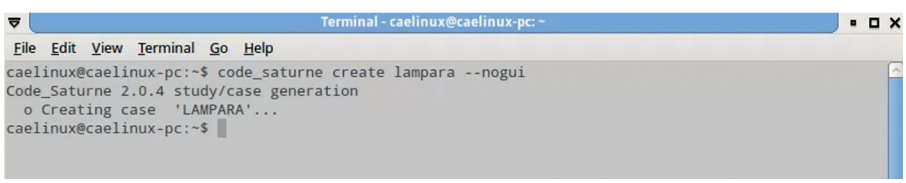


Figura 226. *Abrir una terminal e ingresar comandos*

En este punto nuestro estudio ya estará creado dentro del directorio /home/caelinux/ en donde si enlistamos el contenido, encontraremos un directorio

con el nombre que le indicamos a nuestro estudio anteriormente: LAMPARA. Cabe destacar que el estudio siempre creará las carpetas de los estudios escritos en mayúsculas. Ingresamos a nuestra carpeta LAMPARA, desplegamos su contenido y habrá cuatro carpetas; necesitaremos crear una con el nombre de MESH (en mayúsculas), que podemos observar en la figura 227.

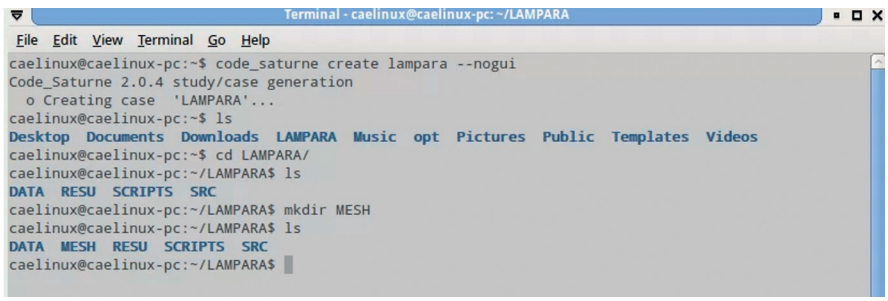


Figura 227. *Directorio MESH*

Una vez creado el directorio MESH podemos realizar le exportación de la malla visto anteriormente, ingresamos a la carpeta MESH una vez exportado la malla con la terminal, listamos el contenido del directorio y encontraremos el archivo exportado con extensión .MED, a continuación, realizaremos un preproceso para verificar que nuestra malla esté correcta; para ello escribimos en nuestra terminal cs_preprocess -m lampara.med --ensight figura 228.

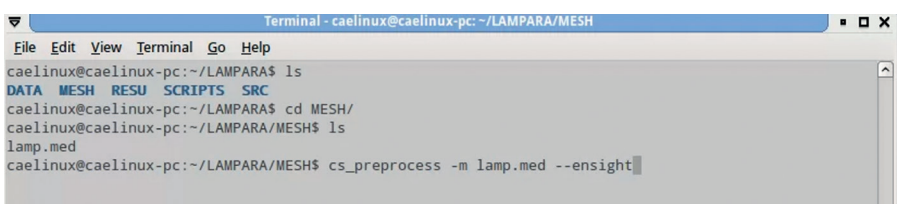


Figura 228. *Preproceso para verificar la malla*

Una vez finalizado el preproceso, nos mostrará un mensaje: Preprocessor finish, el cual indicará que fue correcto. Figura 229.

```
Time and memory summary
-----------------------

  User CPU time                          (sec)   :        1.54
  System CPU time                        (sec)   :        0.09
  Total time                             (sec)   :        1.63
  Total CPU time / Total time                    :        1.00

Memory use summary:

  Total memory used:                                   66.777 Mb
  Theoretical instrumented dynamic memory:             21.633 Mb


  .------------------------.
  |                        |
  |  Preprocessor finish   |
  |                        |
  `------------------------'
```

Figura 229. *Proceso correcto*

Ahora bien, traemos a cuenta la herramienta ParaView con la que podremos visualizar el ejercicio anterior y verificar que no hay problema con la malla. Figura 230.

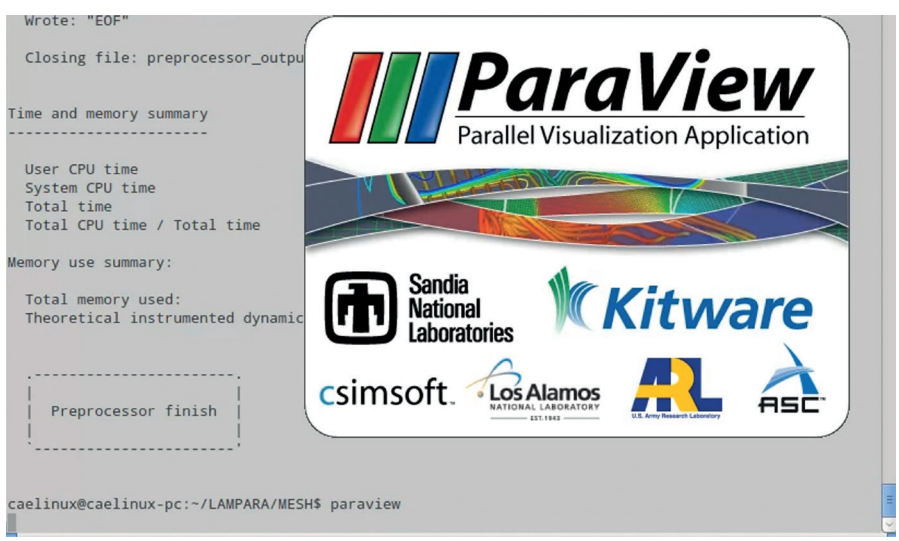


Figura 230. *Herramienta ParaView*

Configuración del estudio

Desde nuestra terminal en el directorio LAMPARA accedemos a DATA, ahí vamos a THCH y enlistamos el contenido.

Nos mostrará unos archivos de los cuales se copiará dp_ELE al directorio DATA así como se observa en la figura 231.

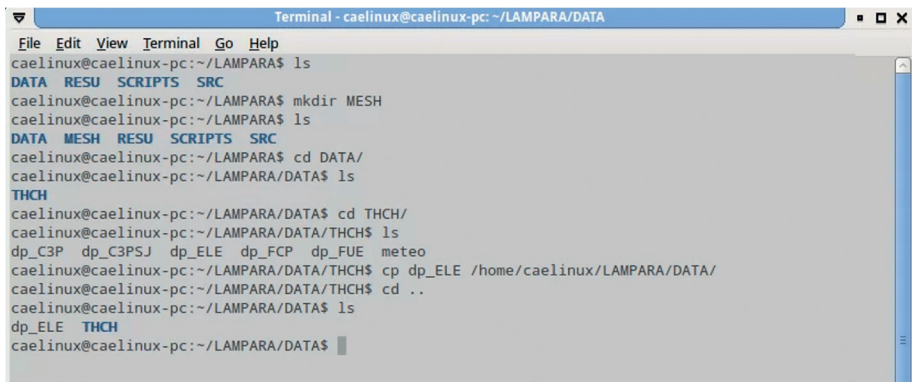


Figura 231. *Directorio LAMPARA*

Accedemos al directorio SRC desde donde copiaremos cinco archivos indispensables para el funcionamiento del estudio. Figura 232.

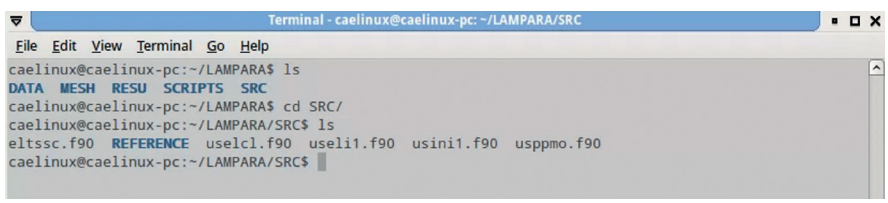


Figura 232 *Directorio SCR*

Figura 233. *Achivos para el funcionamiento del estudio*

Una vez copiados configuraremos el contenido de estos archivos para llevar a cabo nuestro estudio, los cuales veremos más adelante.

Editamos el archivo usini1.f90 con el sistema Leafpad o cualquier otro editor, buscamos la clave t0(iphas) y modificamos la temperatura que estableceremos en 300, como podemos observar en la figura 234.

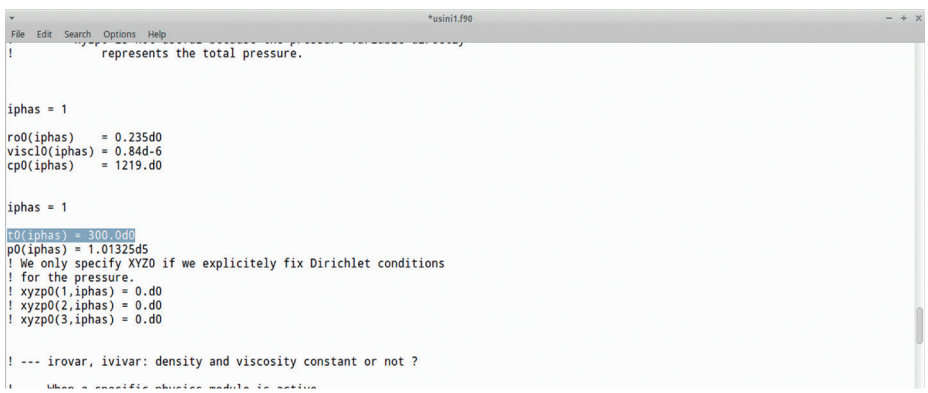


Figura 234. *Archivo usini1.f90*

Buscamos la palabra ntmabs, ahí le indicamos el tiempo que aplicará el estudio que está indicado sobre segundos; en nuestro caso requerimos 120 s. Figura 235.

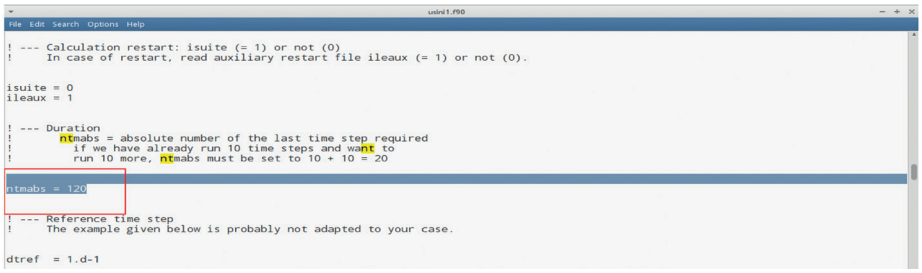


Figura 235. *Tiempo a aplicar*

Buscamos la clave ntchr, en esta opción le indicamos cuántos ciclos queremos que realice la simulación. Figura 236.

Figura 236. *Indicar número de ciclos*

Accedemos al directorio/SCRIPTS/, enlistamos su contenido y ahí encontraremos el archivo con el nombre Runcase que editaremos con Leafpad o con cualquier otro. Figura 237.

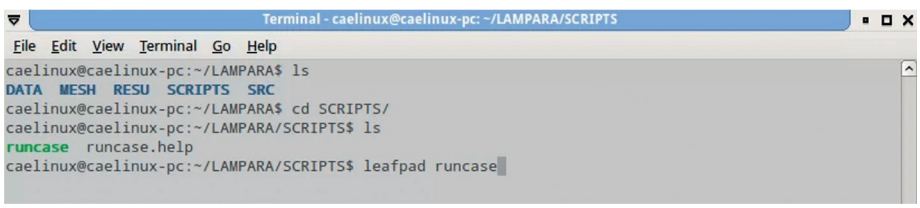


Figura 237. *Directorio /SCRIPTS/*

Ahora realizaremos unas modificaciones a este archivo; como primera modificación agregaremos un valor a la palabra MESH=lamp.med (el cual es

el nombre de nuestra malla). Como segunda modificación agregaremos THERMOCHEMISTRY_DATA=dp_ELE según se indica en la figura 238.

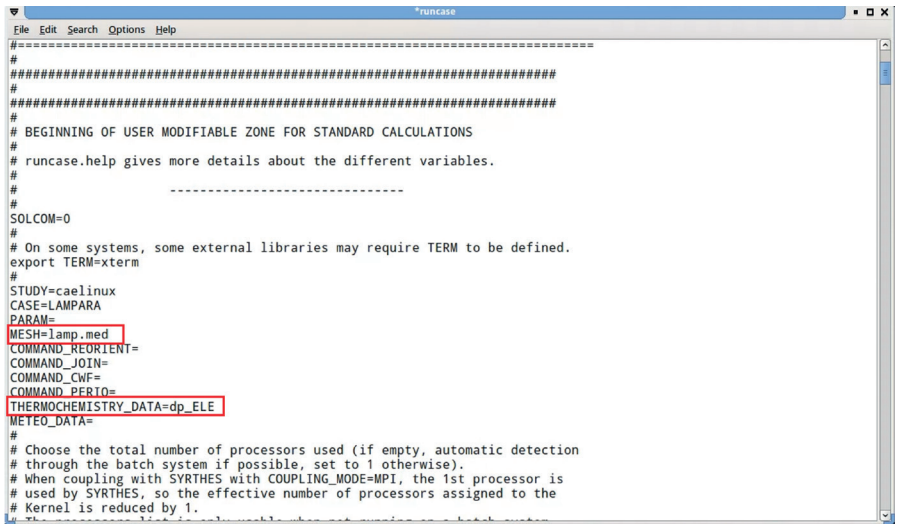


Figura 238. *Modificaciones al archivo*

La tercera modificación es sobre NUMBER _OF_PROCESSORS=4 (número de procesadores a utilizar para la simulación). Figura 239.

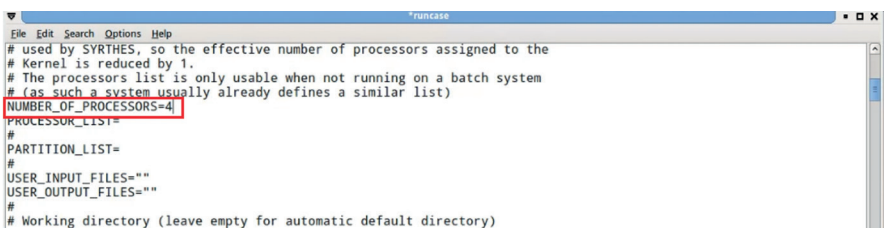


Figura 239. *Número de procesadores a utilizar*

La última modificación será sobre MESHDIR=$CASEDIR/../MESH a MESHDIR=$CASEDIR/MESH como se aprecia en la figura 240.

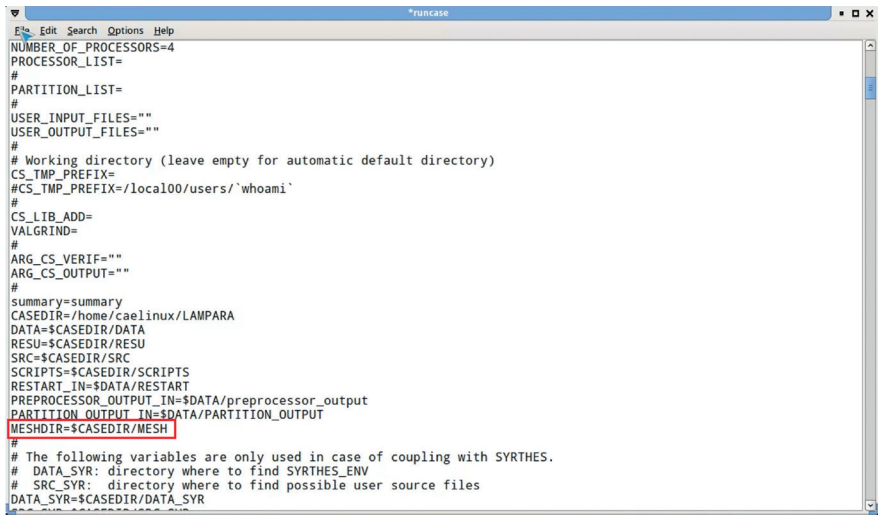


Figura 240. *Última modificación*

Una vez realizadas todas las modificaciones antes vistas, guardaremos los cambios y estamos listos para realizar la simulación de nuestra figura con la ayuda de nuestro programa CODE_SATURNE.

Ejecución de la simulación a nuestra figura

Para que Code Saturne realice el estudio deberemos ejecutar en archivo Runcase, en este caso accedemos sobre el directorio RESU/ ejecutamos el archivo Runcase ubicado en el directorio SCRIPTS/. Lo ilustraremos en la figura 241.

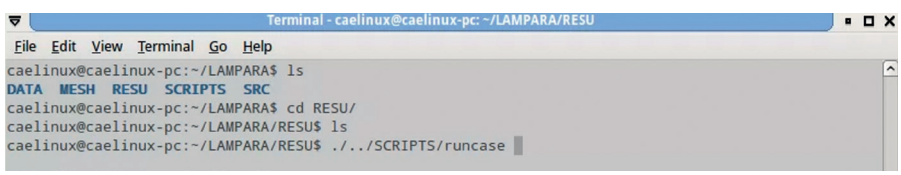


Figura 241. *Estudio a ejecutar*

Deberemos esperar a que el programa realice el proceso. El tiempo que tarde dependerá del número de cálculos que este llevará a cabo. Figura 242.

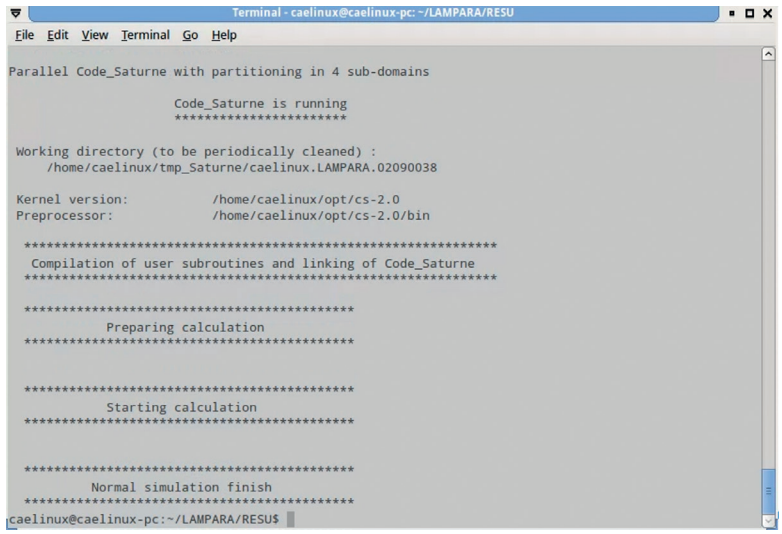


Figura 242. *Espera del proceso*

Una vez finalizado el procesamiento podemos enlistar el contenido del directorio RESU/, observaremos los archivos que ha creado nuestro estudio. Figura 243.

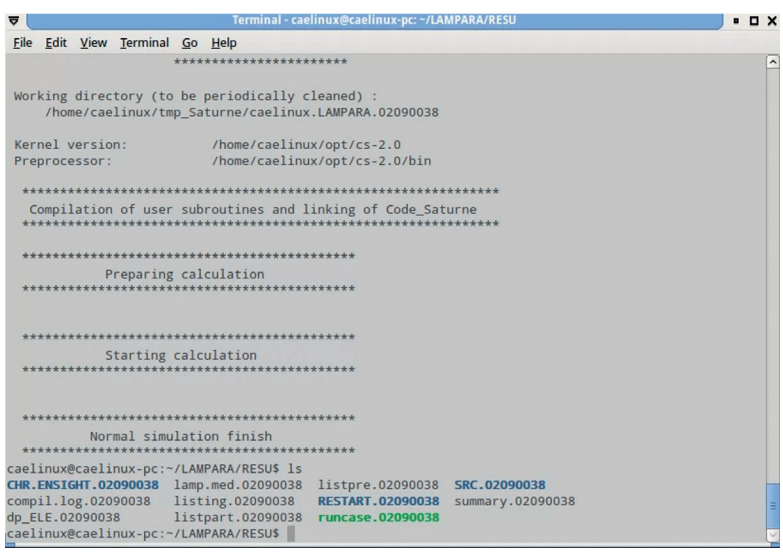


Figura 243. *Directorio RESU*

A continuación deberemos observar el resultado con la herramienta ParaView, la cual ingresaremos en nuestra terminal, como puede apreciarse en la figura 244.

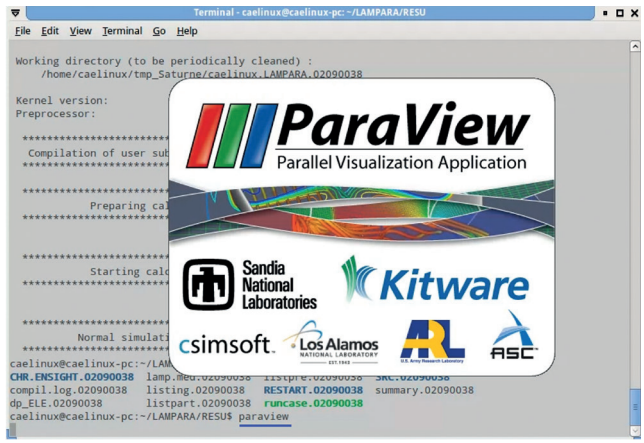


Figura 244. *Resultado con la herramienta ParaView*

Visualización de resultados

Dentro de ParaView abrimos el archivo CHR.case, ubicado en el directorio /RESU/CHR.ENSIGHT.02090038/ y elegimos aplicar (Apply). Figura 245.

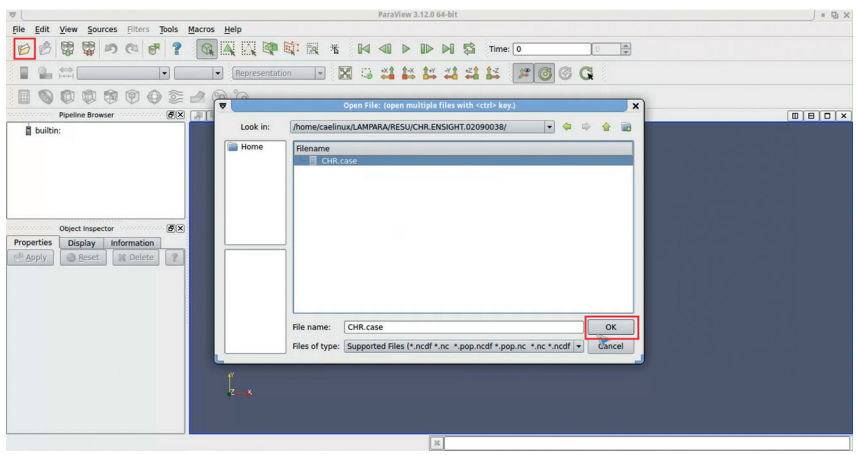


Figura 245. *Directorio /RESU/CHR.ENSIGHT.02090038/*

Realizamos un corte (Slice) y lo aplicamos para que nos aparezca una figura 2D. Figura 246.

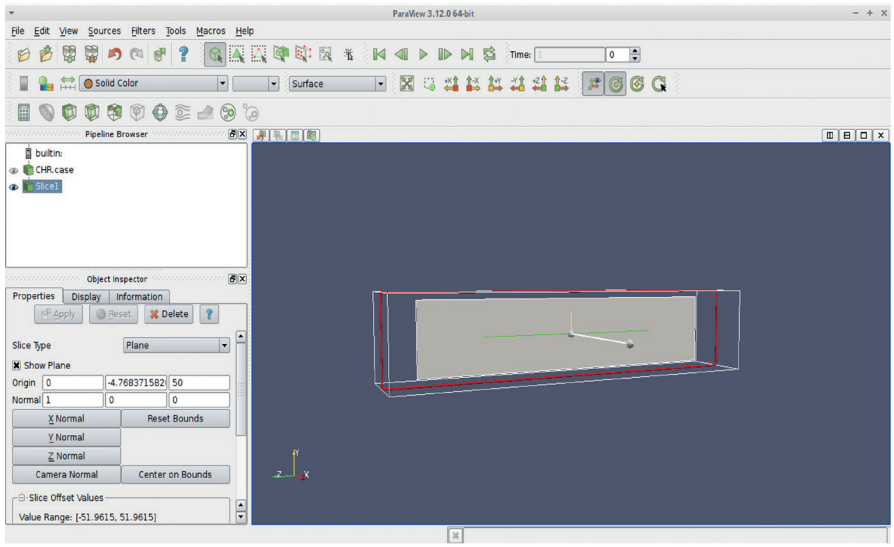


Figura 246. *Figura 2D*

Aplicamos el filtro Cell data to point data en el menú Alphabetical. Figura 247.

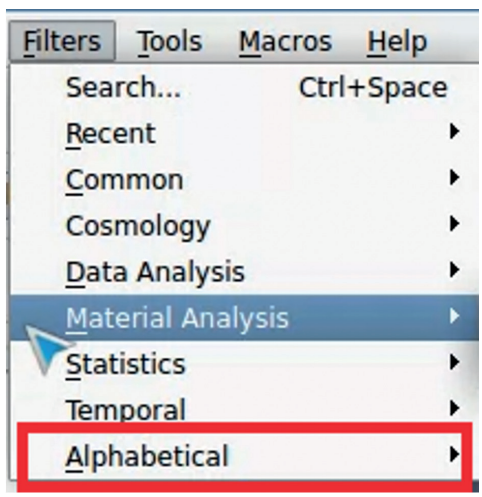


Figura 247. *Filtro Alphabetical*

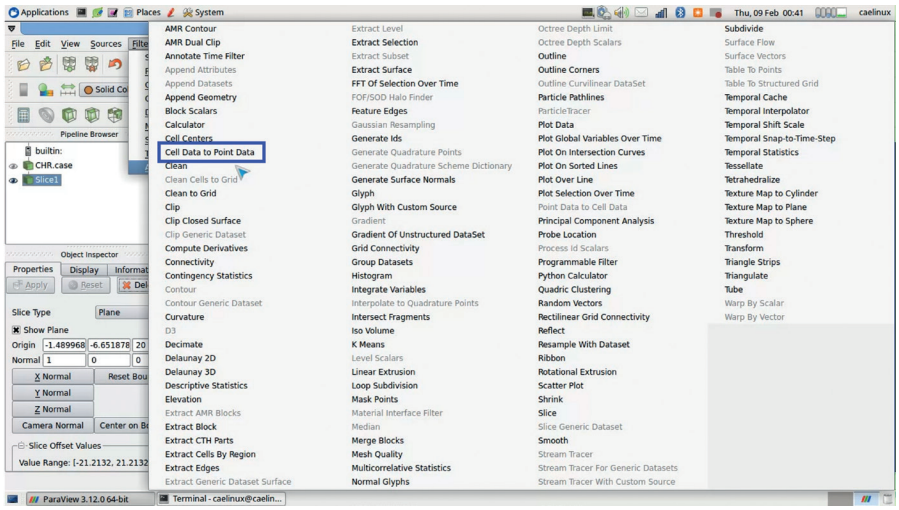


Figura 248. *Cell Data to Point*

Elegimos aplicar. Figura 249.

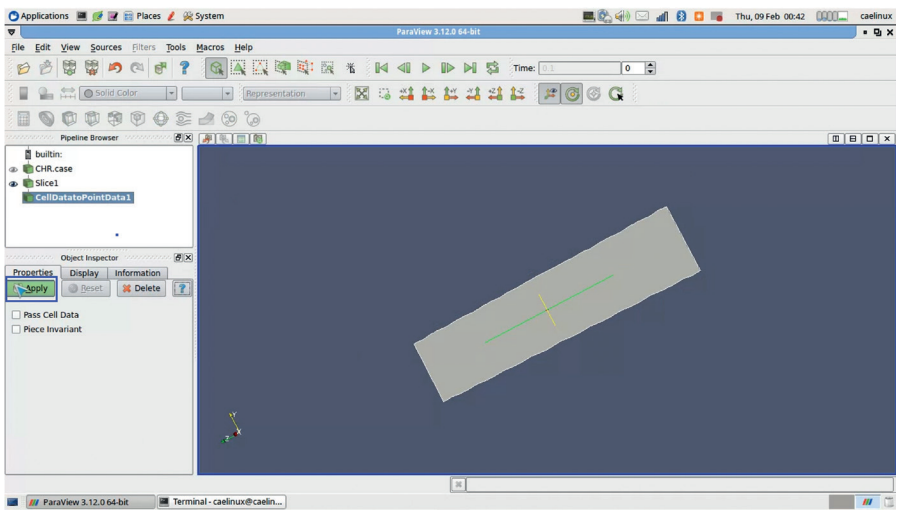


Figura 249. *Aplicamos Cell Data to Point*

Elegimos el estado en el que vamos a trabajar la temperatura (Temper) porque es la forma de representación de la electricidad. Figura 250.

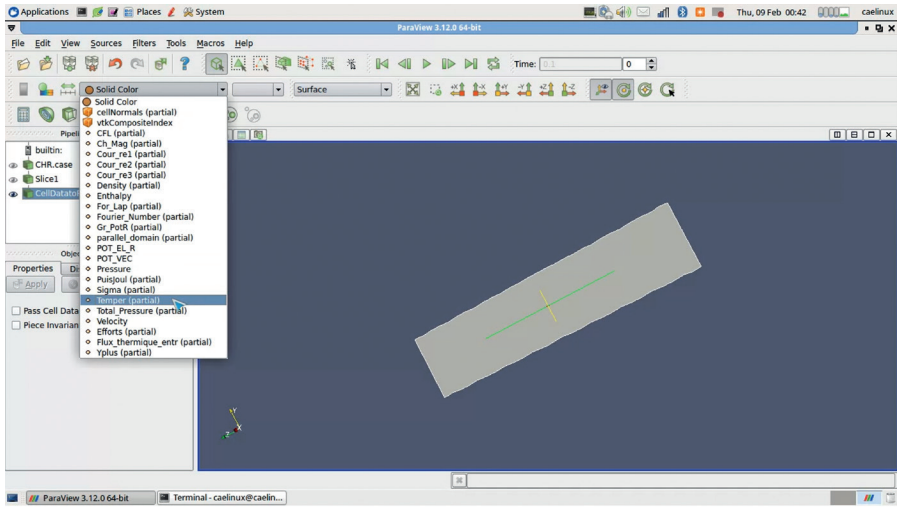


Figura 250. *Estado a trabajar: temperatura*

Mostramos la leyenda de colores para representar la temperatura. Figura 251.

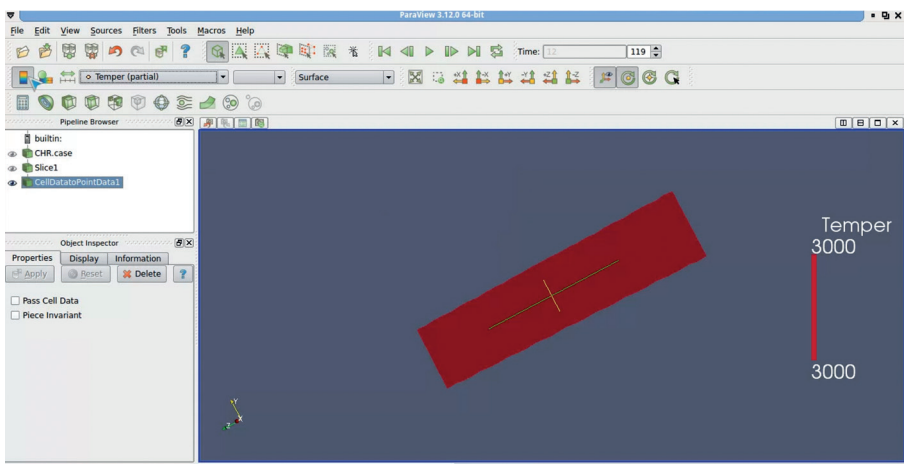


Figura 251. *Leyenda de colores*

Ahora reescalamos el rango de datos que nos mostrará durante la simulación de nuestro plasma. Figura 252.

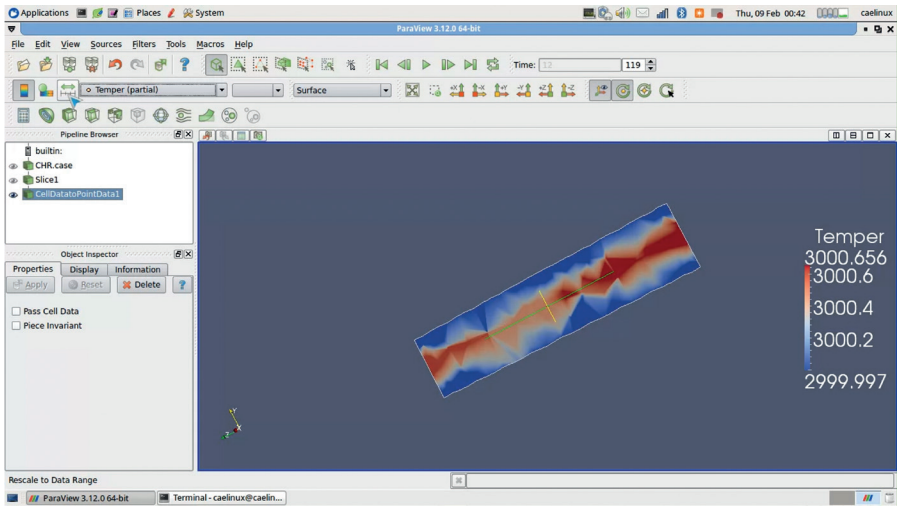


Figura 252. *Simulación plasmada*

A continuación damos clic en Play para correr la simulacion y observaremos los resultados obtenidos en la figura 253.

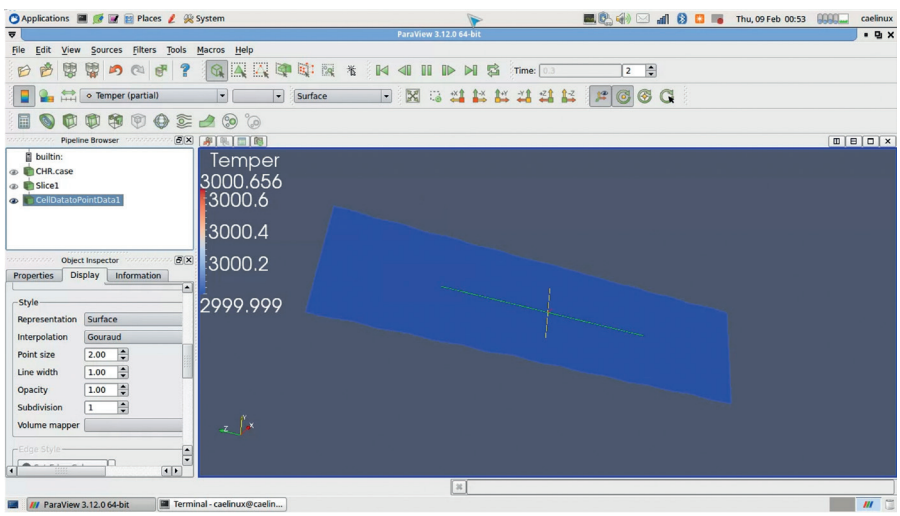


Figura 253. *Simulación corrida 1*

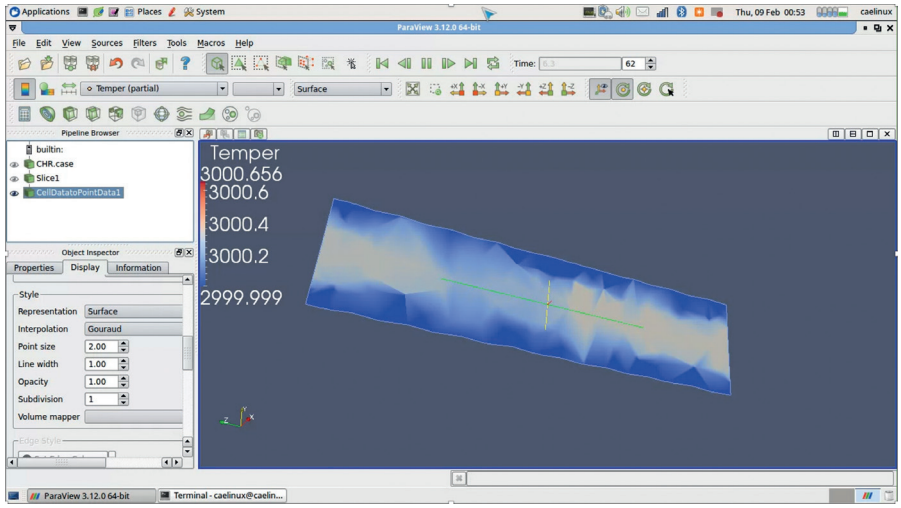


Figura 254. *Simulación corrida 2*

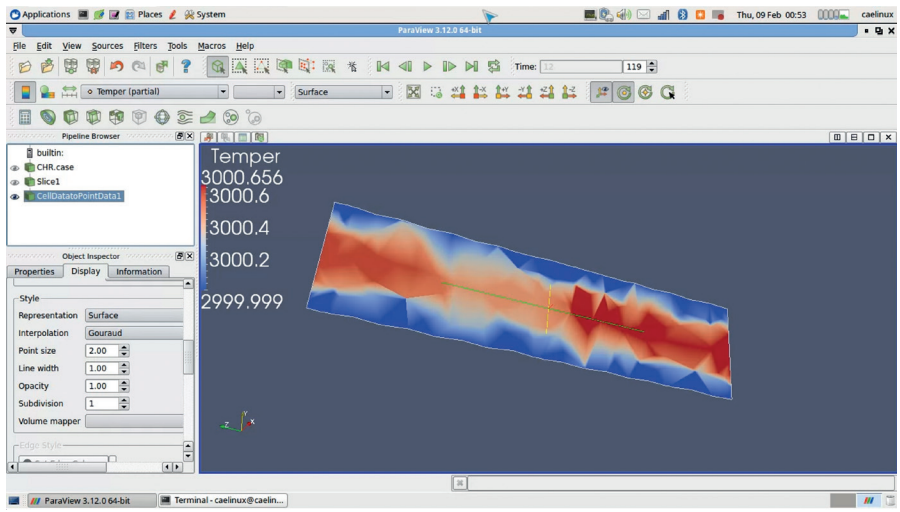


Figura 255. *Simulación corrida 3*

De la misma manera podemos hacer una simulación más realista si nos dirigimos al apartado Pipeline Browser, activamos nuestro objeto CHR.case sobre el ícono del ojo que aparece del lado izquierdo; sobre este objeto nos dirigimos al apartado Object inspector; enseguida Display y nos desplazamos hacia abajo en la opción Opacity, indicamos una transparencia de 0.40;

a continuación reproduciremos nuevamente la simulación y se observarán los resultados más adelante. Figura 256.

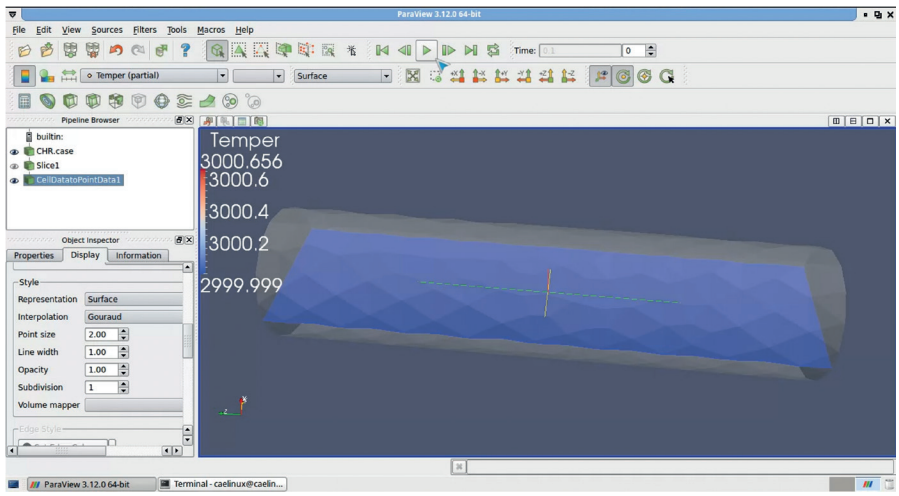


Figura 256. *Simulación realista 1*

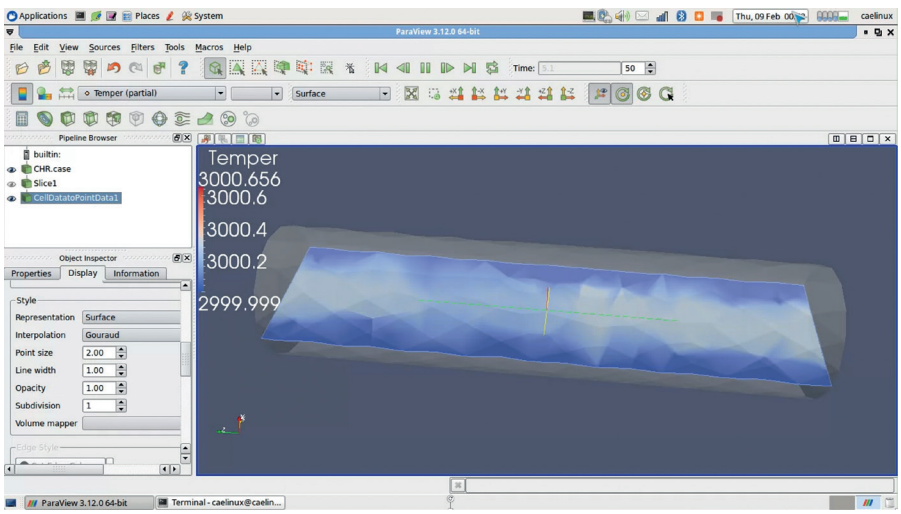


Figura 257. *Simulación realista 2*

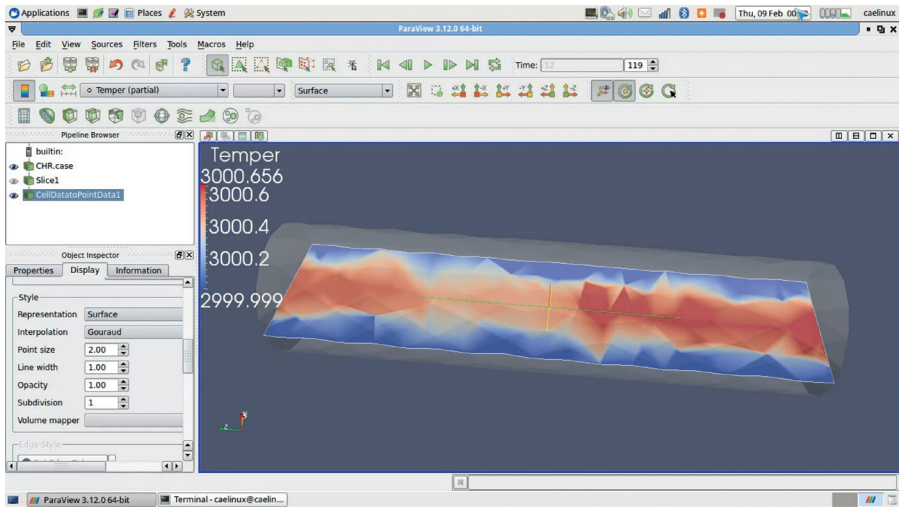


Figura 258. *Simulación realista 3*

Desde esta sección podremos modificar los parámetros sobre la temperatura, los ciclos y los segundos para someter las diferentes condiciones que se requieran.

Como hemos planteado en los diferentes ejemplos, la aplicación de software libre es compatible en la elaboración de geometrías básicas y temperaturas como el plasma frío para la fabricación previa de objetos. Como reflexión final, podemos decir que los programas resultan flexibles, siempre y cuando el operador tenga conocimientos básicos de física, de química y, por supuesto, de programación.

Aunque los pequeños ejercicios realizados han sido heterogéneos y no necesariamente estaban relacionados en la conformación de un solo objetivo, es importante mencionar que el lector sí puede combinar los tres programas a los que hemos dedicado atención: Salome Meca, Code Saturne y ParaView, en un intento por obtener resultados propiciatorios con la realidad.

Reflexiones finales

En las páginas precedentes intentamos crear el ABC de Salome Meca para el desarrollo del Diseño Asistido por Computadora (CAD) y Code Saturne, a través de una sencilla aplicación dentro de la gama infinita de posibilidades del sistema libre, se mostró el uso para la simulación numérica de seis objetos, entre ellos una lámpara de neón. Durante el ejercicio se trató de llevar al lector por el desarrollo de la simulación y la obtención de los resultados de manera muy simple, demostrando que el software libre no es tan complejo como suele creerse, pero sí cuenta con algunas limitantes, aunque no son las relativas al desarrollo de los programas, sino a su funcionalidad.

No existe frontera en las aplicaciones libres, salvo las que se llegan a imponer a voluntad de personas físicas y morales, en tanto que, en realidad, el único inconveniente del código abierto es el que antepone la piratería a las habilidades e imaginación de los usuarios y a la monopolización de los grandes consorcios.

Hay tres áreas de uso de programa libre: en la academia, en la investigación y en la industria. La que corresponde a la academia cuenta con las mayores posibilidades de exploración de este software, debido a que existen en las redes homólogos privados iguales o con mejor potencial que suelen usar los estudiantes; entre ellos destacan Matlab, Autocad, Solidworks y Ansys académico, por citar algunos. Los equiparables en el mundo de los programas libres serían Octave, Freedcad, Code Saturne, Salome Meca y ParaView; localizables para las plataformas Linux y Windows.

En el presente libro hemos observado que una de las características de los sistemas abiertos es la posibilidad, no solo de manipularlos sin restricción económica, sino también de aplicar mejoras, aunque para diversos ingenieros esta posibilidad representa una contradicción al momento de intentar usar la herramienta, por el hermetismo que puede llegar a representar para muchos usuarios.

Personas del área académica, pese a conocer la paquetería libre, consideran que es mucho más complicada, al momento de intentar obtener resultados, en comparación con el software privado que comúnmente usan; ello se debe a que están pasando por un periodo de adaptabilidad que, en un instante, la falta de costumbre puede confundirse con un obstáculo.

A través de pequeños procedimientos como los vistos en páginas anteriores, académicos o estudiantes, poco a poco pueden acumular experiencia, hasta volver indispensables todas las aplicaciones disponibles. En la academia se tiene tiempo para adquirir el hábito y paulatinamente, conforme vayan avanzando los estudiantes en el curso de sus carreras, estos van habituándose hasta desarrollar habilidades, como la programación, la configuración del sistema al gusto del usuario sin restricciones y la creación de nuevos proyectos, por mencionar algunos.

Con un código privado estas áreas de oportunidad no son factibles porque este sistema cuenta con las indicaciones precisas sin posibilidades de ejercitar la destreza de los usuarios. Por tanto, los programas abiertos podrían ser una herramienta indispensable en las universidades, tanto públicas como privadas.

Otro beneficio es el combate a la piratería; sin embargo, la industria es un área donde prácticamente es nulo el uso de programas abiertos. Esto debido a que en muchas ocasiones no hay tiempo para la adaptación de los nuevos usuarios por cuestiones de requerimientos, por lo que normalmente en la industria se usan licencias privadas comerciales. Para los desarrollos de nuevos productos, los sistemas libres no cumplen con los requerimientos necesarios para el desarrollo de nuevos productos por el hecho de ser "abiertos", es decir tienen códigos que no son confiables para la industria.

Por ejemplo, si una empresa que desarrolla y manufactura motores eléctricos y que usa simulación numérica como parte de su proceso de manufactura para la planeación de nuevos productos, normalmente recurre a

programas privados, esto es porque en la mayoría de los sistemas de paga que se ocupan para estos dispositivos, como son los motores eléctricos, les garantizan resultados. El hecho de garantizar una prospectiva alude a que se corroboró la experimentación; se tiene fiabilidad, lo cual no sucede con programas libres que están sujetos a cambios de librerías y con ellos solo es posible verificar a través de la prueba real misma.

Un ejemplo de la escasa o nula aplicación de software libre es en la industria automotriz, en donde para el CAD del diseño de automóviles, se cuenta con normativas que especifican los esquemas de fabricación, dentro de sus diferentes componentes, incluida la prospectiva. La norma ISO 26262 define los requisitos que deben cumplir las funciones relevantes de seguridad del sistema, así como los procesos, métodos y sistemas necesarios. Desafortunadamente la norma no permite software libre.

La norma ISO 26262 solicita pruebas y evaluaciones basadas en su documentación de desarrollo y solicita los resultados en un informe técnico donde se encuentran la evaluación de los sistemas, hardware, software y herramientas utilizadas; por tanto no es posible incluir un programa libre, solo se pueden localizar privados como CATIA para el CAD, Ansys para las simulaciones numéricas tanto para el área electromagnética, mecánica y dinámica de fluidos computacional.

En el área automotriz, existe la prueba de compactibilidad electromagnética para verificar que todos los dispositivos electrónicos de los vehículos sean electromagnéticamente compatibles y no interfieran con los sistemas externos, es decir que puedan convivir sin afectarse entre ellos. Esta prueba la desarrollan tanto en forma física, a través de una cámara anecoica, y en forma simulada haciendo uso del software de simulación de licencia privada.

Para tales pruebas, en la actualidad, la mayoría de las empresas se basan en la simulación numérica, indispensable, más que en las pruebas físicas, a razón de que las cámaras anecoicas son de muy elevado costo.

Hay otras normas aplicadas en la industria que también condicionan el software libre, de manera que el escaso tiempo de adaptación de los usuarios para conocer estos programas hacen que definitivamente sean limitados en el sector industrial, al menos por el momento.

La cuarta revolución industrial, esto es la Industria 4.0 (I4.0), en el contexto mundial dan esperanza las aplicaciones libres, ya que estas práctica-

mente se encuentran conformadas por sistemas ciberfísicos, internet de las cosas, redes, por mencionar algunos; todos estos componentes que pertenecen a la I4.0, por el momento no precisan u obligan a ocupar software privado, por lo que puede existir una pléyade de aplicaciones libres; esto permite que el código libre se vuelva una alternativa importante en las aplicaciones industriales dentro de la I4.0, pero este escenario se ve factible solo en un futuro, aunque todavía lejano.

Por último, las aplicaciones científicas, al igual que la academia, son de las áreas que considero con mayores oportunidades por varios factores, pero uno fundamental es que en la investigación ya se encuentran desarrollados diferentes programas que facilitan su uso, como Code Saturne que es el ejemplo mostrado en este libro, donde para poder realizar el ejercicio de la lámpara de neón con aplicaciones de paga se hubiera requerido específicamente Ansys, en específico Ansys Fluent y Ansys Electronics (Ansys Maxwell), los cuales aparte de ser muy costosos, requieren un periodo de adaptabilidad y experiencia para su uso, y por supuesto el alto costo monetario que se necesita para adquirir este software privado, obteniendo un resultado muy similar en ambos casos, tanto en el libre como en el de paga.

La diferencia solo radica en la calibración de la simulación libre respecto a la privada; como ya lo había mencionado, Ansys soporta los resultados que ofrece el programa abierto, mientras que en un sistema libre se tendría que calibrar la simulación. Esto alude a comparar los resultados obtenidos en la simulación respecto a los resultados obtenidos experimentalmente para asegurar que son correctos. Calibrar la simulación podría tornarse complejo, pero una vez realizada, se tiene la certeza y por tanto las aplicaciones, por complicadas que parezcan, como aquellas que se mostraron en el libro, ofrecerán la seguridad de sus resultados.

Referencias

[1]Alarcón, C. M. D. (2019, 5 septiembre). Software libre y gratuito para la simulación en Ciencias e Ingeniería. linkedin. Recuperado 9 de junio de 2022, de https://es.linkedin.com/pulse/software-libre-y-gratuito-para-la-simulaci%C3%B3n-en-ed%C3%ADaz-alarc%C3%B3n-

[2]CALCULIX: A Three-Dimensional Structural Finite Elemente Program. (s. f.). CALCULIX. Recuperado 9 de junio de 2022, de http://www.calculix.de/

[3]Overview. (s. f.). OpenFOAM. Recuperado 9 de junio de 2022, de https://www.openfoam.com/governance/overview

[4]FlexSim. (s. f.). Manufacturing Simulation. Recuperado 9 de junio de 2022, de https://www.flexsim.com/es/manufacturing-simulation/

[5]FlexSim. (2020, 11 noviembre). Industria 4.0. Recuperado 9 de junio de 2022, de https://www.flexsim.com/es/industria-4-0/

[6]FlexSim. (2022, 25 febrero). Digital Twin. Recuperado 9 de junio de 2022, de https://www.flexsim.com/digital-twin/

[7]FlexSim. (2021, 7 julio). PLC Emulation. Recuperado 9 de junio de 2022, de https://www.flexsim.com/es/plc-emulation/

[8]Metso Outotec. (s. f.). HSC Chemistry. Recuperado 9 de junio de 2022, de https://www.mogroup.com/es/portafolio/hsc-chemistry/#:%7E:text=HSC%20Chemistry%20de%20Metso%20Outotec%20te%20ayuda%20a,caracter%C3%ADsticas%20vers%C3%A1tiles%20de%20procesamiento%20qu%C3%ADmico%2C%20termodin%C3%A1mico%20y%20mineral.

[9]Siemens Digital Industries Software. (2021, 30 noviembre). Solid Edge para equipos y maquinaria industrial | Software de diseño. Solid Edge. Recuperado 9 de junio de 2022, de https://solidedge.siemens.com/es/industries/industrial-machinery/

[10]Code Aster. (s. f.)). Code Aster. Analysis of Structures and Thermomechanics for Studies Research. Recuperado 9 de junio de 2022, de https://www.code-aster.org/spip.php?rubrique2

[11]HELYX-OS GUI for OpenFOAM | ENGYS. (s. f.). Engys. Recuperado 9 de junio de 2022, de https://engys.com/products/helyx-os

[12]LibreCAD - Free Open Source 2D CAD. (s. f.). LibreCAD. Recuperado 9 de junio de 2022, de https://librecad.org/

[13]Deingenierias.com. (2020, 23 julio). Qué es FreeCAD y para qué sirve. Recuperado 9 de junio de 2022, de https://deingenierias.com/software/que-es-freecad-y-para-que-sirve/

[14]Ingeniería libre. (2016, 28 febrero). SALOME Platform: una plataforma para análisis numérico. Recuperado el 10 de junio de 2018, de https://ingenierialibreyabierta.blogspot.com/2016/02/salome-platform-una-plataforma-para.html

Sobre los autores

Mario Ibañez Olvera
Doctor en Ciencias en Ingenieria en Electrónica por el Instituto Tecnológico de Toluca y el Instituto Nacional de Investigaciones Nucleares. Trabajó en Grupo S.S.C. S.A. de C.V., empresa líder en simulación en México, ha sido catedrático de las licenciaturas en Ingeniería Biomédica y Ciencias de la Computación, en la Facultad de Medicina y en la Unidad Académica Profesional Tianguistenco (donde continúa impartiendo clases), ambas de la Universidad Autónoma del Estado de México. En la misma institución ha desarrollado dos estancias posdoctorales. Actualmente es profesor investigador del Tecnológico de Estudios Superiores de Tianguistenco (TEST) y pertenece al Sistema Nacional de Investigadores, Nivel I.
ORCID: https://orcid.org/0000-0003-2828-5389

Alien Blanco Flores
Graduada como Doctora en Ciencias Ambientales (2016) por la Facultad de Química de la Universidad Autónoma del Estado de México. Máster en Ciencias y Tecnología de los Materiales por el Instituto de Ciencia y Tecnología de los Materiales (IMRE) de la Universidad de La Habana, Cuba. Es profesora investigadora de tiempo completo en la División de Ingeniería Mecánica, Tecnológico de Estudios Superiores de Tianguistenco. Pertenece al Sistema Nacional de Investigadores, Nivel I, del Conacyt. Es árbitro de artículos científicos en las revistas: *U. D. C. A Actualidad & Divulgación Científica* de la Universidad de Ciencias Aplicadas y Ambientales U. D. C. A, Bogotá, Colombia, y de la Jour*nal of Environmental Chemical Engineering*, de Elsevier, por citar algunas.
ORCID: https://orcid.org/0000-0003-4035-0550

Simulación de objetos.
Aplicaciones por Salome Meca, Code Saturne y
ParaView, de la autoría de Mario Ibañez Olvera y
Alien Blanco Flores, publicado por la Universidad Autónoma del Estado de México y Ediciones Comunicación Científica S. A. de C. V., se publicó en formato PDF, Epub3 y HTML5, en enero de 2023 en acceso libre.

El conocimiento y aplicación de los modelos de simulación electrónica son todavía incipientes en México, de manera que los autores de *Simulación de objetos. Aplicaciones por Salome Meca, Code Saturne y ParaView* presentan un recorrido básico por este tipo de programación, a través de sencillos y cotidianos ejercicios que son el resultado de la experimentación de los investigadores.

Aun cuando existe una alta demanda en el uso de estos *software* en diferentes ámbitos sociales, muy pocos estudiosos tienen acceso a los programas por su oneroso perfil. Para hacer frente a esta necesidad, los autores consideran que una alternativa la representan los *software* libres. Éstos tienen las funciones para emular cualquier tipo de cuerpo y cuentan con la oportunidad para las áreas académicas. Este libro es una línea básica para observar uno de los terrenos fértiles de la electrónica, aún por explorar.

Mario Ibañez Olvera es Doctor en Ciencias en Ingenieria en Electrónica por el Instituto Tecnológico de Toluca y el Instituto Nacional de Investigaciones Nucleares. Profesor de asignatura de la UAP Tianguistenco, Universidad Autónoma del Estado de México.

Alien Blanco Flores es Doctora en Ciencias Ambientales. Universidad de La Habana, Cuba. Profesora investigadora de tiempo completo en la División de Ingeniería Mecánica, Tecnológico de Estudios Superiores de Tianguistenco (TEST).

COMUNICACIÓN CIENTÍFICA PUBLICACIONES ARBITRADAS
HUMANIDADES, SOCIALES Y CIENCIAS
www.comunicacion-cientifica.com

DOI.ORG/10.52501/CC.095

ISBN UAEMEX 978-607-633-562-8
ISBN ECC 978-607-59473-8-9