



UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS  
FACULTAD 4

## COMPONENTE CON FUNCIONALIDADES PARA ACELERAR EL MODELADO DE TORNILLOS EMPLEANDO LOS CONTENIDOS DE UNA BASE DE DATOS

**Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas**

Autor: Eliany Betancourt Lumpuy

Tutor: Dr.C. Augusto César Rodríguez Medina

**La Habana, 2019**

*La sabiduría no es un producto de la educación, sino del intento de toda la vida para adquirirla.*  
*Albert Einstein*

---

Dedicatoria

---

*A mis padres y hermana, en especial a mi mamá.*

---

## Agradecimientos

---

*Agradezco a mis padres, por su gran esfuerzo y sacrificio, por su apoyo incondicional, por siempre conducirme por el camino correcto y ayudar a convertirme en la mujer que soy hoy, llena de sueños y aspiraciones. A mi hermanita que de una forma u otra aunque ella no lo supiera me llenó con su dulzura y amor, momentos de tristeza. A mi abuela Azucena, por comprenderme siempre y hacerme reflexionar, cuando no he actuado bien, a mi tío Adniel y mi abuelo Pedro, por ayudarme con esta ardua tarea. A mi novio Leonardo, por estar siempre presente, en los momentos más difíciles, por brindarme todo su amor y cariño y hacer que mi estancia en esta universidad fuera más placentera. A mis suegros Leonardo y Mirna, por aceptarme como parte de su familia y por haberme brindado su apoyo de una forma sincera. A todos mis tíos, primos y demás familiares que me apoyaron en estos cinco años. A mi tutor por apoyarme en la realización de este trabajo. A todas mis compañeras de cuarto, desde ese primer año tan duro hasta quinto. A Arlet, mi primera amiga en esta universidad, junto a la cual pasé momentos duros y momentos de alegrías y la que ha estado ahí hasta el día de hoy para mí. A Andy por ayudarme dedicadamente en esta tarea tan dura. A mis compañeros de proyecto Machin, Mario y Eduardo por colaborar con la realización de este trabajo y hacerme pasar buenos momentos. A todas mis amistades y profesores que de una forma u otra contribuyeron a mi formación como profesional. A todos muchas gracias.*

---

## Declaración de autoría

---

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los \_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

---

Eliany Betancourt Lumpuy  
Autor

---

Dr.C. Augusto César Rodríguez Medina  
Tutor

Debido a que el modelado de tornillos es uno de los principales paradigmas en la disciplina del Diseño Asistido por Computadora (CAD por sus siglas en inglés) y la necesidad del grupo de investigación Soluciones Informáticas para la Ingeniería y la Industria de no contar con un componente que permita el modelado de tornillos. Se realizaron una serie de funcionalidades que permiten acelerar el proceso de modelado de tornillos, a través de los contenidos de una Base de Datos (BD) para el sistema de diseño “Ingeniero”. Las funcionalidades de dicho componente se desarrollaron en el lenguaje C++, utilizando como framework de desarrollo Qt y la tecnología OpenCascade para el modelado de sólidos, entre otras que hacen posible que el componente pueda ser utilizado en la industria nacional. Se utilizó como metodología de desarrollo la AUP (Proceso Unificado Ágil), en su variante UCI (Universidad de las Ciencias Informáticas), específicamente el escenario 4, ya que posee un negocio muy bien definido y el cliente estará siempre acompañando al equipo de desarrollo. También se le hicieron varias pruebas de calidad donde se evidencia que el componente presenta algunos errores pero que no son tan graves como para impedir el proceso de modelado de tornillos.

**Palabras clave:** tornillos, Diseño Asistido por Computadora (CAD), Base de Datos.

<b>Introducción</b>	<b>1</b>
<b>1 Fundamentación Teórica</b>	<b>5</b>
1.1 Aspectos preliminares sobre la situación problemática y el proceso de investigación . . . . .	5
1.1.1 Generalidades de los tornillos . . . . .	5
1.2 Algunos aspectos de las normas internacionales que rigen el modelado de los tornillos . . . . .	14
1.3 Sistemas comerciales de diseño asistido por computadora con componentes para el modelado de tornillos a partir de catálogos de piezas normalizadas. . . . .	17
1.4 Conclusiones parciales . . . . .	20
<b>2 Descripción de la propuesta de solución</b>	<b>21</b>
2.1 Metodología de desarrollo . . . . .	21
2.2 Descripción de la propuesta de solución . . . . .	23
2.3 Especificación de requisitos del software . . . . .	24
2.3.1 Requisitos Funcionales . . . . .	24
2.3.2 Requisitos No Funcionales . . . . .	25
2.4 Historias de usuario . . . . .	25
2.5 Análisis y diseño . . . . .	36
2.5.1 Estilo y patrón arquitectónico del software . . . . .	36
2.5.2 Patrones de Diseño . . . . .	38
2.6 Diagrama de clases del diseño . . . . .	38
2.7 Conclusiones del Capítulo . . . . .	40
<b>3 Implementación y pruebas</b>	<b>41</b>
3.1 Implementación . . . . .	41
3.1.1 Detalles técnicos de la implementación . . . . .	41
3.1.2 Estándar de codificación . . . . .	42
3.1.3 Resultados de la implementación . . . . .	42
3.2 Pruebas . . . . .	42
3.2.1 Niveles de prueba . . . . .	47

3.2.2	Métodos de prueba . . . . .	47
3.2.3	Diseño de Casos de Prueba . . . . .	48
3.2.4	Resultados obtenidos de las pruebas funcionales . . . . .	50
3.2.5	Resultados obtenidos de las pruebas de integración . . . . .	51
3.2.6	Pruebas de Aceptación . . . . .	51
3.3	Conclusiones del Capítulo . . . . .	52
<b>Conclusiones</b>		<b>53</b>
<b>Recomendaciones</b>		<b>54</b>
<b>Referencias bibliográficas</b>		<b>55</b>
<b>Apéndices</b>		<b>57</b>
<b>A</b>	<b>Anexos</b>	<b>58</b>
A.1	Casos de prueba . . . . .	58
A.2	Manual de usuario . . . . .	64
A.2.1	Configuración inicial . . . . .	64
A.2.2	Subsistemas o Módulos . . . . .	65



---

## Índice de figuras

---

1.1	Representación del plano inclinado del tornillo. . . . .	6
1.2	Partes del tornillo. . . . .	6
1.3	Cabeza hexagonal . . . . .	7
1.4	Cabeza hexagonal con pivote . . . . .	8
1.5	Cabeza hexagonal con brida . . . . .	8
1.6	Cabeza hexagonal con extremo en punta . . . . .	9
1.7	Cabeza ranurada y ranurada cruciforme . . . . .	9
1.8	Cabeza cuadrada . . . . .	10
1.9	Cabeza cilíndrica con hexágono interior . . . . .	10
1.10	Cabeza de mariposa . . . . .	11
1.11	Cabeza redonda . . . . .	11
1.12	Cabeza de gota de sebo . . . . .	12
1.13	Cabeza torx . . . . .	12
1.14	Diámetro y longitud del tornillo. . . . .	13
1.15	Perfiles de rosca más empleados. . . . .	13
1.16	Paso de rosca. . . . .	14
1.17	Representación de rosca derecha o izquierda . . . . .	15
1.18	Representación de rosca sencilla o múltiple . . . . .	16
1.19	Ejemplo de tornillo según su unión . . . . .	17
1.20	Ejemplo de tornillo según su unión . . . . .	18
1.21	Norma DIN 6921 . . . . .	19
2.1	Fases que propone AUP-UCI . . . . .	22
2.2	Arquitectura de programa principal/subprograma . . . . .	37
2.3	Diagrama de clases del diseño . . . . .	39
3.1	Estándar de codificación . . . . .	42
3.2	Estándar de codificación . . . . .	43
3.3	Estándar de codificación . . . . .	43
3.4	Ejemplo de la normativa <i>CamelCase</i> . . . . .	44
3.5	Funcionalidad buscar . . . . .	44

3.6	Funcionalidad filtrar	45
3.7	Vista en forma de tabla	45
3.8	Diseño de un tornillo	46
3.9	Configuración para la conexión con la Base de datos	46
3.10	Iteraciones de las pruebas	51
A.1	Crear el nuevo visor de la aplicación	64
A.2	Sección de los aceleradores	64
A.3	Configuración para establecer la conexión	65
A.4	Panel de herramientas	65
A.5	Undo y Redo	66
A.6	Search	66
A.7	Filters	66
A.8	TreeView	67
A.9	TableView	67
A.10	List	67
A.11	Modelar el tornillo	68

---

## Índice de tablas

---

2.1	Historia de usuario # 1	26
2.2	Historia de usuario # 2	27
2.3	Historia de usuario # 3	28
2.4	Historia de usuario # 4	29
2.5	Historia de usuario # 5	30
2.6	Historia de usuario # 6	30
2.7	Historia de usuario # 7	31
2.8	Historia de usuario # 8	31
2.9	Historia de usuario # 9	32
2.10	Historia de usuario # 10	33
2.11	Historia de usuario # 11	34
2.12	Historia de usuario # 12	34
2.13	Historia de usuario # 13	35
2.14	Historia de usuario # 14	36
3.1	Diseño de Casos de Prueba RF 1	48
3.2	Diseño de Casos de Prueba RF 9	49
3.3	No Conformidades	50
A.1	Diseño de Casos de Prueba RF №2	58
A.2	Diseño de Casos de Prueba RF №3	60
A.3	Diseño de Casos de Prueba RF №4	60
A.4	Diseño de Casos de Prueba RF №5	60
A.5	Diseño de Casos de Prueba RF №6	61
A.6	Diseño de Casos de Prueba RF №7	61
A.7	Diseño de Casos de Prueba RF №8	61
A.8	Diseño de Casos de Prueba RF №11	62
A.9	Diseño de Casos de Prueba RF №13	63

En el presente trabajo, se exponen los resultados obtenidos en el proceso de investigación realizado, para asegurar el desarrollo de un componente informático, destinado a acelerar el proceso de modelado de tornillos. Dicho componente está inmerso dentro de la disciplina del Diseño Asistido por Computadora (CAD por sus siglas en inglés o DAC por su acrónimo en español, el cual se estará utilizando a lo largo del desarrollo del trabajo). Esta disciplina, mediante algoritmos especializados extrae la información necesaria para conformar la documentación del proyecto, principalmente los planos.

El Diseño Asistido por Computadora abarca el uso de un amplio rango de herramientas computacionales que asisten a ingenieros, arquitectos y a otros profesionales del diseño en sus respectivas actividades. Es todo sistema informático destinado a asistir al diseñador en su tarea específica. El DAC atiende prioritariamente aquellas tareas exclusivas del diseño, tales como el dibujo técnico y la documentación del mismo, pero normalmente permite realizar otras tareas complementarias relacionadas principalmente con la presentación y el análisis del diseño realizado(Oswaldo Rojas Lazo, 2018).

En la actualidad existe una amplia variedad de sistemas informáticos con numerosas funcionalidades para el modelado de piezas mecánicas. Entre ellos se encuentran *Inventor de AutoDesk Incorporated*, *Solid Edge* desarrollado por SIEMENS, *SolidWorks* de *SolidWorks Corp*, entre otros. Estos se caracterizan por ser sistemas comerciales que demandan una licencia para su explotación, lo que imposibilita obtener el código fuente, por lo que no es posible corregir bugs o errores de programación(Oca Montano, s.f.), mientras que en el ámbito del código abierto y *software* libre se encuentran otros como *LibreCAD* desarrollado por *QCad Community Edition* así como *FreeCAD* originado por Juergen Riegel y Werner Mayer.

Varios de estos sistemas contienen sesiones de trabajo que por lo general son para la elaboración de piezas, ensambles y planos, pero el modelado de componentes normalizados poseen “centros de contenidos”o librerías que representan entre si, catálogos de piezas.

Los catálogos de piezas están formados por dos componentes principales: un sistema de Base de datos (BD por su acrónimo en español, el cual se empleará en el desarrollo del trabajo) y un componente para gestionar los datos de las piezas y modelarlos. El componente trabaja como un acelerador de diseño. Los sistemas comerciales mencionado en párrafos anteriores, usan como sistema de BD, *Microsoft SQL* y *Microsoft SQL Server*, mientras que los módulos, para el modelado se implementan con funcionalidades de las librerías gráficas o “*kernels*” como *ACIS(Spatial Releases 2017 1.0, Delivering Technology Enhancements*

*Aimed at Innovation and Industrialization 2016*) y *ParaSolid(ADINA-M 2015)*. Dentro de los componentes normalizados se encuentran los elementos de fijación y entre ellos los tornillos o los reconocidos como pernos. Los tornillos son un elemento mecánico utilizado en la fijación de piezas, que está dotado de una caña o vástago con rosca, que, mediante una fuerza de torsión ejercida en su cabeza con una herramienta adecuada, se puede roscar o desenroscar en un agujero o atravesar las piezas y acoplarse a una tuerca(*Enciclopedia de ciencia y técnica 1984*).

Según lo planteado en párrafos anteriores, el uso de sistemas de diseño comerciales no es una opción viable, ni segura para la industria nacional, ya que son costosos además de que producto al bloqueo económico impuesto por los Estados Unidos, Cuba no los puede adquirir(Oca Montano, s.f.). Una opción viable que ha sido considerada es utilizar herramientas de *software* libre, como por ejemplo el “*LibreCAD*” y el “*FreeCAD*”, o conformar soluciones de código abierto, lo que permitiría desarrollar soluciones informáticas con ciertos niveles de independencia tecnológica. Sin embargo, estas herramientas no cuentan con un componente que facilite el modelado de tornillos.

El presente trabajo se encuentra insertado en las actividades del grupo llamado "Soluciones Informáticas para la Ingeniería y la Industria"(SIPII), donde la dedicación principal es desarrollar las actividades de investigación y desarrollo con el propósito de cumplir las demandas de las actividades de diseño.

En el contexto de lo planteado anteriormente, fue identificada como **situación problemática**, que algunas necesidades importantes del diseño no existen, como por ejemplo:

- No existe un componente que contenga funcionalidades para el modelado de tornillos, pues no están implementadas las funcionalidades para este fin.
- No presenta una implementación de funcionalidades para asignar los contenidos de una base de datos a tornillos.

A partir de la situación problemática descrita anteriormente se plantea como **problema de investigación**: ¿Cómo acelerar el proceso de modelado de tornillos, empleando los contenidos de una base de datos?.

**El objeto de estudio** de la investigación se centra en el modelado de tornillos, empleando los contenidos de una base de datos, teniendo como **campo de acción**, el desarrollo de un componente para acelerar el proceso de modelado de tornillos.

**El objetivo general** de este trabajo de diploma es: crear un componente con funcionalidades para acelerar el modelado de tornillos, empleando los contenidos de una base de datos.

Para dar cumplimiento al objetivo general se cumplieron las siguientes tareas de investigación:

#### Fase de análisis

- Búsqueda y revisión bibliográfica sobre el objeto de investigación, lo que incluye el estudio del estado del arte sobre sistemas, para el diseño y modelado de piezas, en aplicaciones de diseño asistido por

computadoras, así como la asimilación de los conceptos y tecnologías requeridos para el proceso de desarrollo.

- Fundamentación de las necesidades y requerimientos de proyecto que avalan el desarrollo del componente.
- Obtención de requisitos a partir de las fuentes disponibles (información disponible en publicaciones científico-técnicas y foros especializados colocados en Internet) y entrevistas con el usuario (jefe de proyecto, especialistas de ingeniería y diseño).
- Estudios de los aspectos de la metodología de software empleada en el proyecto.

#### Fase de síntesis

- Definición de la estructura del componente a partir de la arquitectura existente para la aplicación.
- Elaboración y presentación de la propuesta, obtención de conclusiones, resultados, definición de los procedimientos para el proceso de desarrollo.
- Determinación de la estructura de clases del componente.
- Diseño de las interfaces gráficas requeridas para el componente.
- Integración del componente en la aplicación del proyecto.

#### Fase de pruebas

- Pruebas al componente (de las diferentes funcionalidades y de integración).

#### Métodos teóricos

**Análisis y síntesis:** se empleó para la construcción y desarrollo de la teoría, para la profundización en el tema y la sistematización del conocimiento.

**Modelado:** se empleó en la generación de los artefactos que permitan lograr una mejor comprensión entre el equipo de desarrollo y las personas relacionadas.

#### Métodos empíricos

**Observación:** se empleó para caracterizar las soluciones, teniendo en cuenta potencialidades y carencias, de otras aplicaciones DAC (SolidWorks, AutoCAD Inventor), y así establecer los requisitos con las principales funcionalidades de estos.

**Experimentación:** se empleó para comprobar que el modelado sea el adecuado, a partir de la forma del objeto y las dimensiones del mismo.

El documento está estructurado por tres capítulos:

En el Capítulo 1 se refleja la fundamentación teórica, abordando los conceptos fundamentales y los sistemas similares.

En el Capítulo 2 se realiza la descripción de la solución propuesta mediante una especificación de requisitos ya sean funcionales como no funcionales. Se realiza una propuesta de solución a partir de los conceptos identificados y las relaciones entre ellos, aplicando metodologías, tecnologías, herramientas y patrones de diseño para su realización. Luego de su realización se integrará el componente en la aplicación del proyecto “Ingeniero”.

En el Capítulo 3 se realizan las diferentes pruebas de funcionalidad y de integración al sistema, garantizando que el componente cumpla con las expectativas del cliente.

En el presente capítulo se aborda algunos aspectos teóricos de los elementos de fijación y dentro de ellos los tornillos, aplicando el diseño asistido por computadora.

### **1.1. Aspectos preliminares sobre la situación problemática y el proceso de investigación**

Los sistemas comerciales poseen compuestos para el modelado de piezas normalizadas empleando los datos contenidos en una Base de datos. Los tornillos son un ejemplo claro de lo planteado, ya que poseen ciertas y determinadas características, y en la mayoría de los casos estas se encuentran en una BD en forma de tabla, dando lugar a que la información sea almacenada permanentemente. Estos compuestos dan lugar a la generación de planos en forma digital, proceso que antes se hacía manualmente. Esta actividad de diseño ha constituido un avance para el sector de la ingeniería, la arquitectura y la construcción permitiendo incorporar cambios con facilidad.

El DAC ha proporcionado una herramienta relativamente simple para la visualización en 3D, proceso que antiguamente se basaba en técnicas de dibujo lentas y laboriosas, que además no eran interactivas. Muestra el proceso completo de fabricación de un determinado producto con todas y cada una de sus características como tamaño, contorno, etc. Todo esto se graba en la computadora en dibujos o diseños bidimensionales o tridimensionales. Así si el creador puede con posterioridad mejorarlos, o compartirlos con otros para perfeccionar su diseño(*Áreas Técnicas s.f.*).

#### **1.1.1. Generalidades de los tornillos**

Aunque algunos historiadores mencionan que ya los antiguos egipcios hacían uso del tornillo, el primer registro que se tiene acerca de su invención se remonta al nombre del griego “Arquitas de Tarento” que viviera entre los años 430 a 360 antes de “Cristo”, Arquitas fue también quien inventó la poléa. Posteriormente



“Arquímedes”, perfeccionó el tornillo e inventó el torno y un “tornillo sin fin”, el cual usó para trasladar agua a superficies más alta(A.Enriquez.Hz, s.f.).

El tornillo es un elemento que deriva directamente del plano inclinado y trabaja asociado a un orificio (CEJAROSU, 2006), ver (Figura 1.1).

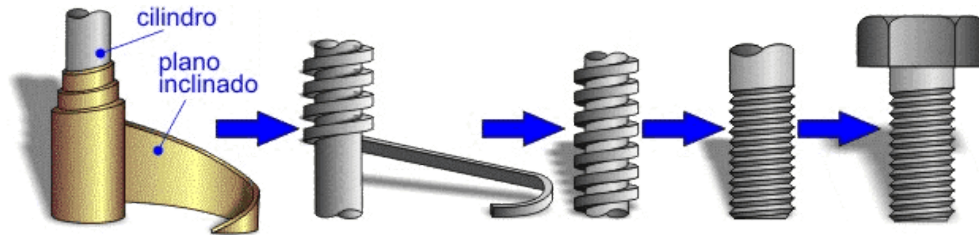


Figura 1.1. Representación del plano inclinado del tornillo.

Básicamente puede definirse como un plano inclinado enrollado sobre un cilindro, o lo que es más realista, un surco helicoidal tallado en la superficie de un cilindro (si está tallado sobre un cilindro afilado o un cono tendremos un tirafondo).

### Partes de un tornillo

En él se distinguen tres partes básicas: cabeza, cuello y rosca, ver (Figura 1.2):



Figura 1.2. Partes del tornillo.

1. La cabeza permite sujetar el tornillo o imprimirle un movimiento giratorio con la ayuda de útiles adecuados.
2. El cuello es la parte del cilindro que ha quedado sin roscar (en algunos tornillos la parte del cuello que está más cercana a la cabeza puede tomar otras formas, siendo las más comunes la cuadrada y la

nervada).

3. La rosca es la parte que tiene tallado el surco. Además cada elemento de la rosca tiene su propio nombre; se denomina filete o hilo a la parte saliente del surco, fondo o raíz a la parte baja y cresta a la más saliente.

### Características de los tornillos

Todo tornillo se identifica mediante 5 características básicas: cabeza, diámetro, longitud, perfil de rosca y paso de rosca(CEJAROSU, 2006), ver (Figuras 1.3, hasta la 1.15).

1. La **cabeza** permite sujetar el tornillo o imprimirle el movimiento giratorio con la ayuda de útiles adecuados (Los más usuales son llaves fijas o inglesas, destornilladores o llaves *Allen*). Las más usuales son la forma hexagonal o cuadrada, pero también existen otras como semiesférica, gota de sebo, cónica o avellanada, cilíndrica, entre otras, ver (Figuras 1.3, hasta la 1.13).

A continuación se describen algunos tipos de cabeza de tornillo(Angeles Virguez, 2015):

- Cabeza hexagonal: Este es uno de los tornillos más utilizados. Tal como lo indica su nombre, tiene una cabeza en forma de hexágono y suele utilizarse para la fijación o el montaje de piezas e incluso para presión, ver (Figura 1.3).

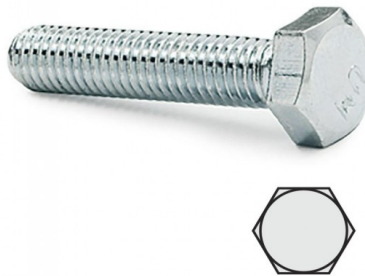


Figura 1.3. Cabeza hexagonal

- Cabeza hexagonal con pivote: Este tornillo, además de permitir uniones con gran apriete, ofrece la posibilidad de inmovilizar la unión utilizando un pasador en el pivote, ver (Figura 1.4).
- Cabeza hexagonal con brida: Este tipo de tornillo también permite aplicar un gran apriete. Pero su ventaja es que no es necesario usar una arandela entre su cabeza y la pieza que se va a unir, ver (Figura 1.5).
- Cabeza hexagonal con extremo en punta (tornillo prisionero): Se trata de una varilla que se enrosca por uno o dos de sus extremos. Es conocido como tornillo prisionero, porque impide el movimiento entre las piezas unidas, ya que se coloca entre la tuerca y el tornillo. Se utiliza para el ensamblaje, ver (Figura 1.6).



Figura 1.4. Cabeza hexagonal con pivote



Figura 1.5. Cabeza hexagonal con brida

- Cabeza ranurada y ranurada cruciforme (phillips): Estos tornillos son utilizados cuando no se necesita aplicar un gran apriete. Presentan una ranura recta en la superficie de su cabeza que se corta perpendicularmente, ver (Figura 1.7).
- Cabeza cuadrada: Al igual que los tornillos de cabeza hexagonal, este tipo de tornillo también se utiliza en casos donde se necesita aplicar un gran apriete como en la fijación de herramientas de corte, ver (Figura 1.8).
- Cabeza cilíndrica con hexágono interior: También llamados *Allen*, estos tornillos cuentan con un hueco hexagonal en la cabeza para encajar una llave *Allen*. Son piezas cilíndricas que se utilizan en uniones donde se necesitan grandes aprietes y que además resulten estrechos, ver (Figura 1.9).
- Cabeza de mariposa: Este tipo de tornillos se utiliza para las uniones que no necesitan de un gran apriete. Además, suelen estar sometidas a montajes y desmontajes manuales frecuentes, ver (Figura 1.10).



Figura 1.6. Cabeza hexagonal con extremo en punta



Figura 1.7. Cabeza ranurada y ranurada cruciforme

- Cabeza redonda: Este tipo de tornillo cuenta con una cabeza con forma esférica y con una base plana, ver (Figura 1.11).
  - Cabeza de gota de sebo: Este es un tornillo que cuenta con una superficie de apoyo con forma cónica y con una cabeza ligeramente ovalada, ver (Figura 1.12).
  - Cabeza Torx: Se trata de un tornillo con un hueco en la cabeza que tiene forma de estrella, ver (Figura 1.13).
2. El **diámetro** es el grosor del tornillo medido en la zona de la rosca. Se suele dar en milímetros(CEJAROSU, 2006), ver (Figura 1.14).
  3. La **longitud** del tornillo es lo que mide la rosca y el cuello juntos, ver (Figura 1.14).
  4. El **perfil de rosca** hace referencia al perfil del filete con el que se ha tallado el tornillo; los más empleados son(ibíd.), ver (Figura 1.15): Las roscas en “V ”aguda suelen emplearse para instrumentos de precisión (tornillo micrométrico, microscopio...); la *Witworth* y la métrica se emplean para sujeción



Figura 1.8. Cabeza cuadrada



Figura 1.9. Cabeza cilíndrica con hexágono interior

(sistema tornillo-tuerca); la redonda para aplicaciones especiales (las lámparas y portalámparas llevan esta rosca); la cuadrada y la trapezoidal se emplean para la transmisión de potencia o movimiento (grifos, presillas, gatos de coches...); la dientes de sierra recibe presión solamente en un sentido y se usa en aplicaciones especiales (mecanismos dónde se quiera facilitar el giro en un sentido y dificultarlo en otro, como tirafondos, sistemas de apriete...).

5. El **paso de rosca** es la distancia que existe entre dos crestas consecutivas. Si el tornillo es de rosca sencilla, se corresponde con lo que avanza sobre la tuerca por cada vuelta completa. Si es de rosca doble el avance será igual al doble del paso, ver (Figura 1.16). Es importante aclarar que según el perfil de la rosca se define el tipo de rosca. Los más comunes para sujeción son *Withworth* y métrica. Estos tipos de rosca están normalizados, lo que quiere decir que las dimensiones de diámetro, paso, ángulo del filete, forma de la cresta y la raíz, etc... ya están predefinidas.

La rosca métrica se nombra o designa mediante una M mayúscula seguida del diámetro del tornillo (en milímetros). Así, M8 hace referencia a una rosca métrica de 8 mm de grosor. Si el tornillo es métrico de rosca fina (tiene un paso menor del normal), la designación se hace añadiendo el paso a

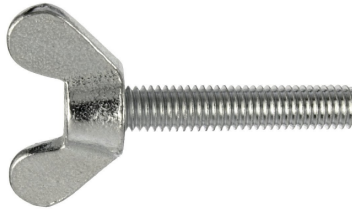


Figura 1.10. Cabeza de mariposa



Figura 1.11. Cabeza redonda

la nomenclatura anterior. Por ejemplo, M20x1,5 hace referencia a un tornillo de rosca métrica de 20 mm de diámetro y 1,5 mm de paso.

#### **Tipos de roscas(CEJAROSU, 2006)**

##### **1. Rosca derecha o izquierda:**

- Según se talle el surco (o, figuradamente, se enrolle el plano) en un sentido u otro tendremos las denominadas rosca derecha (con el filete enrollado en el sentido de las agujas del reloj) o rosca izquierda (enrollada en sentido contrario), ver (Figura 1.17).
- La más empleada es la rosca derecha, que hace que el tornillo avance cuando lo hacemos girar sobre una tuerca o un orificio roscado en el sentido de las agujas del reloj (el tornillo empleado en los grifos hace que estos cierren al girar en el sentido de las agujas del reloj, lo mismo sucede con los tapones o tapas de las botellas de bebidas gaseosas o de agua).

##### **2. Rosca sencilla o múltiple**

- Se pueden tallar simultáneamente uno, dos o más surcos sobre el mismo cilindro, dando lugar a tornillos de rosca sencilla, doble, triple... según el número de surcos tallados sea uno, dos, tres, ver (Figura 1.18).



Figura 1.12. Cabeza de gota de sebo



Figura 1.13. Cabeza torx

- La más empleada es la rosca sencilla, reservando las roscas múltiples para mecanismos que ofrezcan poca resistencia al movimiento y en los que se desee obtener un avance rápido con un número de vueltas mínimo (mecanismos de apertura y cierre de ventanas o trampillas).

### **Funciones de los tornillos**

Los tornillos presentan una función mecánica, la cual enmarca la unión de dos o más piezas entre sí. Esta unión, normalmente fija y desmontable, puede tener lugar por ([Diecnico 2018](#)):

- Apriete: Cuando el tornillo, por medio de su cabeza, ejerce la presión que garantiza la unión entre las piezas.
- Presión: Cuando el tornillo, por medio del extremo de su vástago, presiona contra una pieza y produce su inmovilización.
- Guía: Cuando el tornillo, por medio del extremo de su vástago, asegura una posición determinada entre las piezas, permitiendo, no obstante, cierto grado de libertad. Ver (Figuras [1.19](#) y [1.20](#))

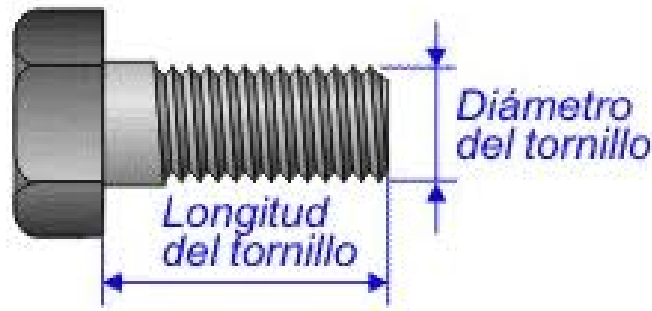


Figura 1.14. Diámetro y longitud del tornillo.



Figura 1.15. Perfiles de rosca más empleados.

Los tornillos presentan una gran utilidad, un ejemplo claro es el sistema tornillo-tuerca, el cual enmarca la unión desmontable de objetos, es decir, cuando se combina el tornillo con una tuerca, este permite comprimir entre esta y la cabeza del tornillo las piezas que se quieran unir. Un ejemplo claro lo podemos evidenciar en motores eléctricos, estanterías desmontables, entre otros(CEJAROSU, 2006).

Otra utilidad de este sistema tornillo-tuerca es empleando como tuerca las propias piezas a sujetar. En este caso es usual que el agujero de la pieza que toca la cabeza del tornillo se taladre con un diámetro ligeramente superior al del tornillo, mientras que la otra pieza (la que hace de tuerca) esté roscada. Se emplea para sujetar chapas (lavadoras, neveras, automóviles, etc.) o piezas diversas (juguetes, ordenadores, etc.) sobre estructuras.

Los tornillos están entre los tipos más comunes y básicos de elementos de sujeción para una amplia variedad de tareas. Vienen en diferentes tamaños y se construyen a partir de diferentes materiales. En comparación con otros elementos de fijación, como clavos, los tornillos tienen varias ventajas. Por ejemplo se pueden desmontar fácilmente, se pueden unir distintos materiales, con distintos tipos de fabricación: compuestos, materiales laminados, tratados térmicamente, así como los costos operativos son bajos.

Los tornillos pueden poseer fallos que podrían ocasionar averías en cualquier ensamblaje. Un ejemplo de fallos que pudieran tener es, un defecto de diseño o de cálculo porque sus dimensiones o calidades no



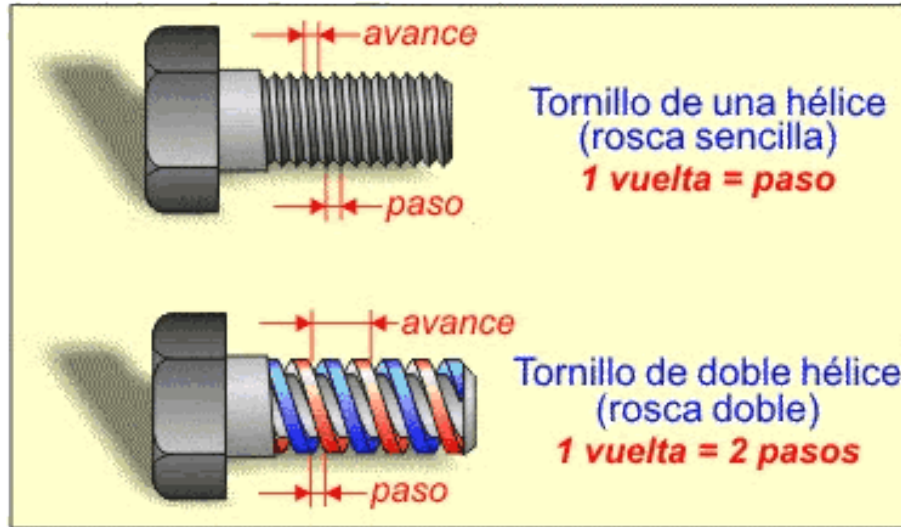


Figura 1.16. Paso de rosca.

sean las adecuadas. Otro defecto sería de fabricación, donde la calidad del material constituyente no sea la prevista en el diseño, o un defecto dimensional en lo que respecta principalmente a las tolerancias que debe tener su roscado. El tercer defecto puede ser un montaje deficiente por no aplicar el par de apriete adecuado, de acuerdo con su calidad y dimensiones. Otro defecto se produciría por deterioro del tornillo si resulta atacado por la oxidación y corrosión si no ha sido protegido debidamente. El último defecto grave que puede tener un tornillo es cuando se procede al desmontaje de un ensamblaje y si por causa de la oxidación y corrosión el tornillo se descabeza en el momento de intentar aflojarlo.

## 1.2. Algunos aspectos de las normas internacionales que rigen el modelado de los tornillos

Durante el siglo XVI surgieron los tornillos de madera, los cuales empiezan a aplicarse en máquinas de guerra y otros artilugios mecánicos, entonces no había ni tornillos ni tuercas iguales, ya que eran fabricados de forma artesanal y variaban tanto en tamaño como en diámetro de la rosca, así como en la separación de la misma. Fue hasta la revolución industrial cuando, con la llegada de las máquinas se comenzó a producir en gran escala, pero continuó el problema de que las medidas eran todas diferentes, ya que no existía una estandarización en la métrica de su fabricación(Casanova, s.f.).

Es entonces cuando el inglés Joseph Whitworth sugiere en el año 1841 un paso de rosca universal para cualquier tornillo, independientemente del fabricante y lugar de origen. A partir de ese momento es cuando todo lo que resulta armado es más confiable, ya que el tornillo da la seguridad que no se tenía con los clavos o grapas, era mucho mas durable y eficiente. Fue desde ese entonces, que empezó lo que hoy relacionamos

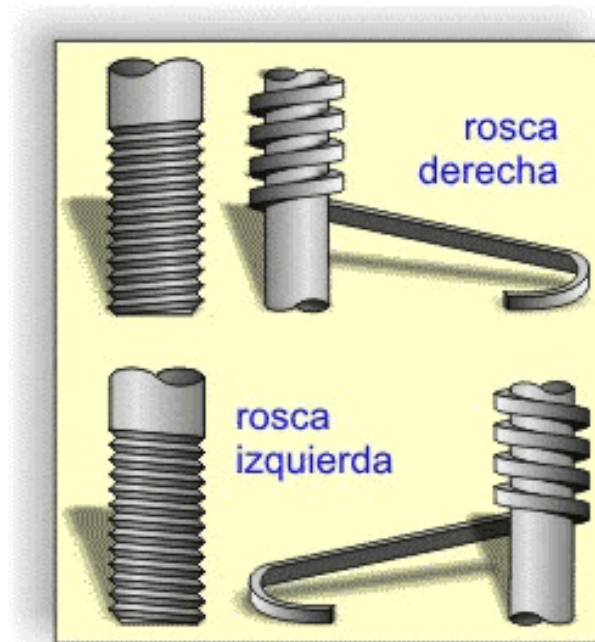


Figura 1.17. Representación de rosca derecha o izquierda

como normalización.

La normalización es el conjunto de prescripciones generales y de reglas que establecieron los países industrializados para favorecer el comercio y racionalizar la fabricación de todo tipo de bienes y servicios. Se refleja en forma de documentos técnicos llamados: especificaciones, legislaciones y normas que permiten especificar, unificar y simplificar. Las especificaciones garantizan las características de los productos fabricados, la unificación facilita el intercambio de elementos de diferente origen y la simplificación disminuye su coste (Búa, 2014).

La normalización es fundamental en el dibujo técnico, ya que permite unificar y simplificar el lenguaje gráfico de representación, acorta el tiempo de dibujo y facilita su interpretación sin equívocos.

En general, el conjunto de normas relativas al dibujo de piezas y conjuntos se puede dividir en tres categorías: de representación, sobre las dimensiones y de designación.

- Normas de representación: codifican el trazado propiamente dicho de una pieza o de un conjunto.
- Normas sobre las dimensiones: se refieren principalmente a las dimensiones de las piezas: medidas nominales, parciales y totales, medidas de tolerancia de fabricación, entre otras.
- Normas de designación: referidas a los elementos de máquinas, que por su gran difusión se normalizaron y estandarizaron mediante un código de identificación: tornillería en general, elementos de transmisión, etc.

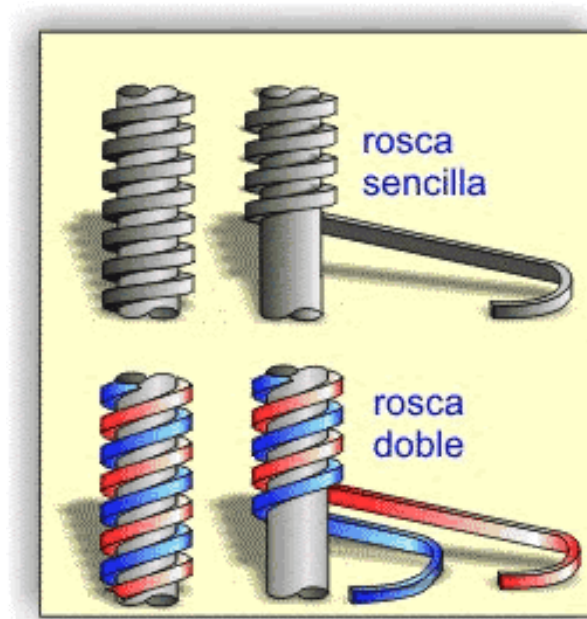


Figura 1.18. Representación de rosca sencilla o múltiple

Beneficios del uso de las normas en la tornillería(*Las normas más habituales en tornillería 2017*).

1. Facilita la creación de productos seguros, fiables y de calidad.
2. Las empresas incrementan su productividad y minimizan errores y gastos.
3. El usuario puede comparar directamente productos de diferentes fabricantes, lo que facilita que nuevas compañías puedan entrar en nuevos mercados y potencia el desarrollo de un comercio global con bases justas.
4. Proporciona seguridad a los consumidores y usuarios finales de dichos productos porque garantiza que los productos certificados se ajustan a las mínimas normas internacionalmente.

Algunas de las entidades normalizadoras más conocidas internacionalmente son: Norma australiana (AS por sus siglas en inglés), norma alemana (DIN por sus siglas en inglés), norma china (GB por sus siglas en inglés), norma india (IS por sus siglas en inglés), norma internacional (ISO por sus siglas en inglés) y la norma japonesa (JIS por sus siglas en inglés).

Un ejemplo claro de un modelado de un tornillo mediante una norma, específicamente la norma DIN 6921, lo evidencia la figura 1.21, donde se ve reflejado una serie de propiedades que posee dicha norma. Ver (Figura 1.21).

Dichas entidades proponen un amplio catálogo de estándares pertenecientes al diseño mecánico que van desde la creación de una rosca, un tornillo, hasta la creación de estructuras y especificaciones físicas de

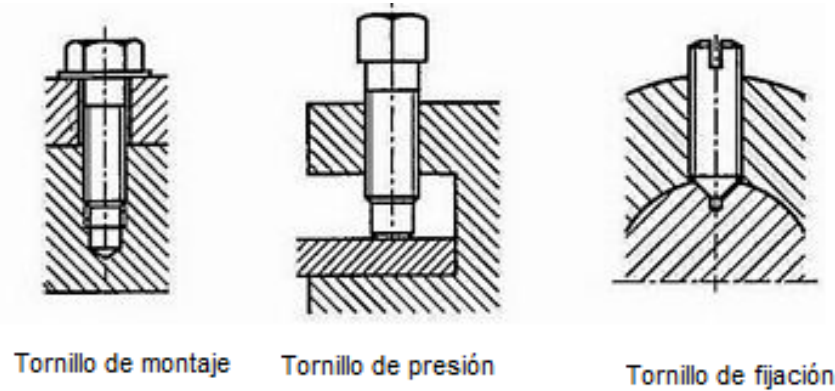


Figura 1.19. Ejemplo de tornillo según su unión

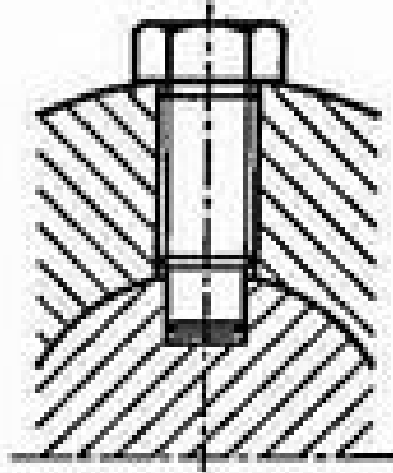
materiales utilizados en la industria. La mayoría de los sistemas DAC existentes en la actualidad se rigen también por estas normas para el diseño mecánico.

### 1.3. Sistemas comerciales de diseño asistido por computadora con componentes para el modelado de tornillos a partir de catálogos de piezas normalizadas.

Se describen los sistemas existentes comerciales ya que no existe ninguno libre que contenga un componente, para el modelado de tornillos.

*Autodesk Inventor* proporciona soluciones de ingeniería y diseño de grado profesional para diseño mecánico en 3D, simulación, visualización y documentación. Con el *software Inventor*, los ingenieros pueden integrar dibujos 2D *AutoCAD* y los datos 3D en un solo modelo digital, la creación de una representación virtual del producto final que les permita validar la forma, el ajuste y la función del producto antes de que se haya construido. *Autodesk Inventor* le permite diseñar, visualizar y simular productos digitalmente, lo que ayuda a reducir los costes de desarrollo, llegar al mercado más rápido, y hacer grandes productos (*Autodesk Inventor Professional 2017. 2018*).

Como modelador paramétrico, no debe ser confundido con los programas tradicionales de DAC. *Inventor* se utiliza en diseño de ingeniería para producir y perfeccionar productos nuevos, mientras que en programas como *Autocad* se conducen solo las dimensiones. Un modelador paramétrico permite modelar la geometría, dimensión y material de manera que si se alteran las dimensiones, la geometría actualiza automáticamente basándose en las nuevas dimensiones. Esto permite que el diseñador almacene sus conocimientos de cálculo dentro del modelo, a diferencia del modelado no paramétrico, que está más relacionado con un “tablero de bocetos digitales”. *Inventor* también tiene herramientas para la creación de piezas metálicas (*ibíd.*).



## Tornillo de guía

Figura 1.20. Ejemplo de tornillo según su unión

Los bloques de construcción cruciales de *Inventor* son las piezas. Se crean definiendo las características, que a su vez se basan en bocetos (dibujos en 2D). También pueden utilizarse los planos de trabajo para producir los bocetos que se pueden compensar de los planos útiles de la partición. La ventaja de este diseño es que todos los bocetos y las características se pueden corregir más adelante, sin tener que hacer de nuevo la partición entera. Este sistema de modelado es mucho más intuitivo que en ambientes antiguos de modelado, en los que para cambiar dimensiones básicas era necesario generalmente suprimir el archivo entero y comenzar de cero.

*Inventor* utiliza formatos específicos de archivo para las piezas (*.IPT*), ensamblajes (*.IAM*), vista del dibujo (*.IDW* y *.DWG*) y presentaciones (*.IPN*), pero el formato del archivo de *AutoCAD* *.DWG* puede ser importado/exportado como boceto.

Las últimas versiones de *Inventor* incluyen funcionalidades que poseían muchos modeladores 3D de mediano y alto nivel. Utiliza el Gestor de Formas (*Shape Manager*) como su *kernel* de modelaje geométrico, el cual pertenece a *Autodesk* y fue derivado del *kernel* de modelaje *ACIS*. Además incluye, en la versión professional, las herramientas necesarias para crear piezas de plástico y sus respectivos moldes de inyección. Cuenta también con análisis de tensiones por elementos finitos y análisis dinámicos. Creación y análisis de estructuras, piping y cableado, y las tecnologías *iPart*, *iAssembly*, *iMates*, *iCopy*, *iLogic* hacen que el diseño sea más rápido y eficiente. Su combinación con *Autodesk Vault* y *Autodesk 360* la hacen líder en el mercado

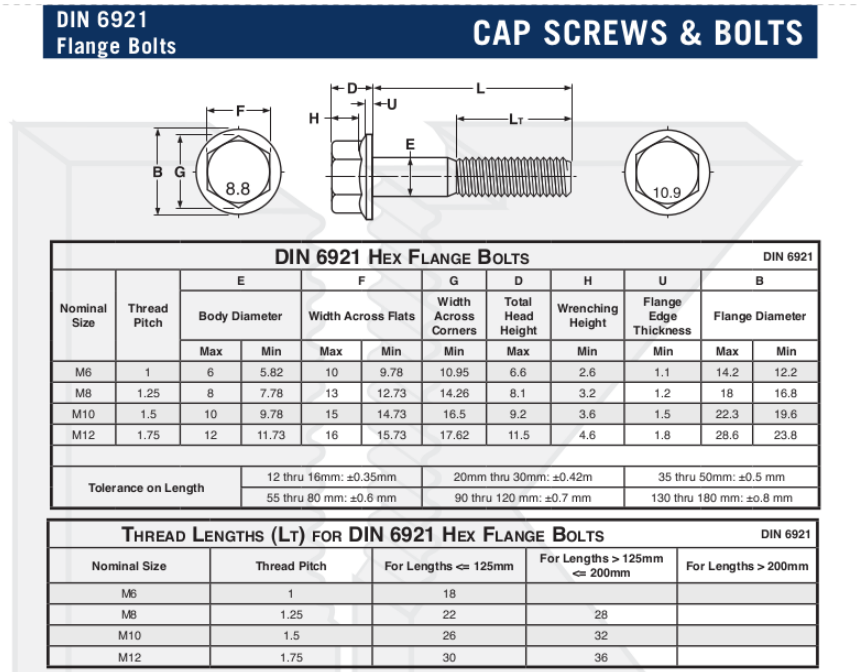


Figura 1.21. Norma DIN 6921

del diseño mecánico(Autodesk Inventor Professional 2017. 2018).

SolidWorks es un software DAC para modelado mecánico en 2D y 3D, permite modelar piezas y conjuntos y extraer de ellos tanto planos técnicos como otro tipo de información necesaria para la producción. Es un programa que funciona con base en las nuevas técnicas de modelado con sistemas de diseño asistido por computadora. El proceso consiste en traspasar la idea mental del diseñador al sistema de diseño, “construyendo virtualmente”la pieza o conjunto. Posteriormente todas las extracciones (planos y ficheros de intercambio) se realizan de manera bastante automatizada(Solidworks Company History 2018).

El módulo para el diseño de componentes normalizados en SolidWorks en su versión del 2017 llamado Assembly específicamente Design Library, permite seleccionar la categoría del componente que queremos modelar, el tipo de norma, la selección de los atributos para el diseño así como su visualización en el plano de trabajo donde se le pueden ir modificando su tamaño.

Solid Edge Standard Parts en su versión 2015, es un poderoso gestor de piezas estándar que permite a los diseñadores definir, almacenar, seleccionar y posicionar piezas utilizadas comúnmente, como elementos de sujeción, rodamientos y perfiles de acero, de forma rápida y eficiente, permitiendo un ensamblado rápido y preciso de conjuntos 3D. El asistente presenta todos los elementos disponibles utilizando un sistema de clasificación de fácil acceso, en el que se incluyen las vistas en 3D y símbolos en 2D de cada pieza. Las vistas en sección, correspondiendo con las normas internacionales, muestran de forma clara a que se refiere cada

parámetro. La base de datos también utiliza un sistema de filtrado inteligente para mostrar únicamente combinaciones de medidas relevantes. Por ejemplo, el diámetro específico de un tornillo mostrara únicamente las longitudes disponibles para ese diámetro(*Solid Edge V15 añade biblioteca de Standard Parts 2018*).

No solo es sencillo encontrar la pieza correcta, además los elementos utilizados con más frecuencia pueden ser marcados como favoritos para su fácil acceso posterior. Con *Solid Edge Standard Parts* se tiene la opción de colocar componentes como piezas simplificadas. La potente tecnología de simplificación utilizada por *Solid Edge*, permite apagar de la visualización de las piezas aquellas operaciones no esenciales, mejorando de esta manera el rendimiento de la visualización de los ensamblajes 3D y las vistas y planos 2D. Esto no elimina las operaciones referidas, y estas estarán disponibles para realizar análisis detallados o renderizados de alta calidad. Adicionalmente, la tecnología “Captura de Ajuste” está disponible en cada pieza estándar. Utilizando piezas que contienen la suficiente inteligencia para saber cómo se tienen que ensamblar.

A pesar de que “Autodesk Inventor”, “SolidWorks”y “Solid Edge”son buenas herramientas a tener en cuenta a la hora del modelado de tornillos, estas no se pueden utilizar, debido a que Cuba no tiene permitido su acceso por ser sistemas considerados de alta tecnología y comerciales. Sin embargo, su estudio permitió conocer funcionalidades como: modelar los tornillos a partir de las distintas normas que existen y las peculiaridades de cada una.

## 1.4. Conclusiones parciales

Al finalizar este capítulo se pude arribar a las siguientes conclusiones:

- A partir de la información consultada se determinó que para el modelado de los tornillos se empleará un surco helicoidal tallado en la superficie de un cilindro, utilizando una representación de una rosca derecha y sencilla, a través de un perfil de rosca del tipo métrica y *whitworth*, por todas las características antes mencionadas.
- El estudio de los sistemas similares permitió identificar que los comerciales, “Autodesk Inventor”, “Solid Edge”y “SolidWorks”, poseen componentes independientes dedicados a acelerar el proceso de modelado de los tornillos, sin embargo los sistemas libres como “FreeCAD”y “LibreCAD”no lo poseen.



---

## Descripción de la propuesta de solución

---

En este capítulo se plantea la propuesta de solución que incluye la descripción de las funcionalidades, los requisitos funcionales y no funcionales, las historias de usuario y aspectos de diseño (el estilo y patrón arquitectónico, el modelo de clases), así como la metodología, herramientas y tecnologías a emplear.

### 2.1. Metodología de desarrollo

Para el desarrollo de la propuesta de solución fue utilizada la metodología “Proceso Unificado Ágil”(Agile Unified Process, AUP por sus siglas en inglés y PUA por su acrónimo en español, el cual se estará utilizando en los siguientes párrafos). Los principios que sustentan PUA son(EDEKI, s.f.):

- La mayoría de la gente no va a leer documentación detallada. Sin embargo, se necesitará orientación y formación de vez en cuando.
- La descripción del proyecto debe ser en unas pocas páginas.
- Se ajusta a los valores y principios descritos en la “Alianza Ágil”.
- El proyecto debe centrarse en ofrecer valor esencial en lugar de características innecesarias.
- Los desarrolladores deben estar libres de utilizar las herramientas más adecuadas para la tarea en cuestión, en lugar de cumplir con un decreto.

En la Universidad de las Ciencias Informáticas (UCI por su acrónimo en español) se realizó una variación a dicha metodología con el propósito de normalizar el proceso de desarrollo de software dentro de todos sus centros productivos. De las 4 fases que propone PUA (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de PUA en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre, ver (Figura 2.1)(Ciencias Informáticas, s.f.).

AUP-UCI también define cuatro escenarios para modelar el sistema en los proyectos productivos(ibíd.):

- **Escenario No 1:** Proyectos que modelen el negocio con Casos de Uso del Negocio solo pueden modelar el sistema con Casos de Uso del Sistema.



Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
Construcción		
Transición		
	Cierre	En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Figura 2.1. Fases que propone AUP-UCI

- **Escenario No 2:** Proyectos que modelen el negocio con Modelo Conceptual solo pueden modelar el sistema con Casos de Uso del Sistema.
- **Escenario No 3:** Proyectos que modelen el negocio con Diagrama de Procesos del Negocio solo pueden modelar el sistema con Diagrama de Requisitos por Proceso.
- **Escenario No 4:** Proyectos que no modelen negocio solo pueden modelar el sistema con Historias de Usuario.

Siguiendo la política de desarrollo de *software* de la institución, para la presente investigación se seleccionó el escenario 4 debido a que posee un negocio muy bien definido, además el cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y se cuenta con poco tiempo para su realización.

## 2.2. Descripción de la propuesta de solución

Para la propuesta de solución se tuvo en cuenta una serie de herramientas y tecnologías que aseguran que el componente funcione como es debido, ejemplo de ellas se tiene la tecnología *OpenCascade* (OCCT, por sus siglas en inglés) es una biblioteca de clases orientada a objetos desarrollada en C++ y diseñada para la producción de sofisticadas aplicaciones de Diseño Asistido por Computadora (CAD), de Fabricación Asistida por Computadora (CAM por sus siglas en inglés) y de Ingeniería Asistida por Computadora (CAE por sus siglas en inglés). Provee servicios para superficies 3D y modelado de sólidos, intercambio de datos CAD y visualización. Cuenta con una comunidad que ha desarrollado *Open CASCADE Community Edition* (OCE por sus siglas en inglés), la cual es reconocida y aceptada por *OPEN CASCADE Company*. OCE es una versión de OCCT desarrollada por las experiencias de la comunidad a través de las recomendaciones de optimización de las versiones liberadas mediante foros o en la propia página principal de desarrollo de esta tecnología (*Open CASCADE Technology 2018*). Se elige el uso del *framework* de desarrollo *QtCreator* en su versión 5.9.5, así como lenguaje de programación C++, por su uso en la tecnología de *OpenCascade* y ser el que se usa actualmente en el proyecto al que está integrada la investigación. Se decide emplear *Visual Paradigm* para UML 8.0 como herramienta de modelado, por propiciar un conjunto de funcionalidades que apoyan al desarrollo de programas informáticos así como soporta el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Debido a las características y a las indicaciones del proyecto se utilizó como sistema de control de versiones al *GitLab*.

A partir de lo anteriormente expuesto sobre las herramientas y tecnologías a emplear, se propone desarrollar un componente con funcionalidades para acelerar el proceso de modelado de tornillos, con el propósito de utilizarlo en la aplicación de diseño asistido por computadora “Ingeniero”. Éste será utilizado por la sección Standard dentro de la sección de aceleradores de diseño *Accelerators*. Al activarse el botón *Standard Part* se muestra una ventana con diferentes opciones que permite establecer la conexión con la base de datos ya creada con el nombre BaseDatos (posee 300 tablas). Antes de ser establecida la conexión el usuario debe registrarse por Admin o Invitado.

1. Opción *Host*: el usuario puede insertar un host local.
2. Opción *DataBase name*: nombre de la base de datos.
3. Opción *username*: nombre del usuario (Admin o Invitado).
4. Opción *password*: contraseña asociada al rol establecido.
5. Opción *Search DataBase*: contiene las base de datos disponibles, son reflejadas en la opción *DataBase name*.
6. Opción *Test Connection*: permite establecer o no la conexión con la base de datos.

Al realizar la configuración de forma correcta se presiona el botón aceptar y aparece la *Application Pro-*

*gramming Interface* (API por sus siglas en inglés) con el nombre *Standard Part Library* que permita al usuario acceder a las distintas opciones que tendrá asociadas el menú, las cuales serán ejecutadas mediante los algoritmos necesarios para su desarrollo. Se enuncian a continuación las opciones asociadas al menú:

- Opción **View (Vista)**: permite visualizar el contenido en forma de árbol (*TreeView*), de tabla (*TableView*) y en forma de lista (*List*).
- Opción **Tools (Herramientas)**: permite filtrar la información de acuerdo a las distintas normas (*Filters*), realizar búsquedas mediante cualquier tipo de norma existente (*Search*), así como incluirá opciones para la navegación hacia delante y hacia atrás (*Undo* y *Redo*).

Al presionar la opción *TreeView* se mostrará un árbol en la parte izquierda de la aplicación, el cual tendrá asociado los distintos tipos de tornillos existentes. Al pulsar cualquiera de los mismos se mostrará en la parte superior derecha al árbol, las diferentes normas asociada al tipo de tornillo seleccionado. Al ser presionada una de las normas existentes se mostrará en la parte inferior derecha al árbol, una tabla con los datos de la misma que están reflejados en la Base de datos. Luego al ser seleccionada una fila correspondiente de la tabla y presionar el botón Aceptar, se mostrará en el visor de la aplicación “Ingeniero” el tornillo seleccionado.

## 2.3. Especificación de requisitos del software

Se pueden clasificar en funcionales y no funcionales. Existen diferentes métodos para la descripción de los requisitos, uno de ellos es el uso de las Historias de Usuario.

### 2.3.1. Requisitos Funcionales

Los requisitos funcionales son “las declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a las entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer”(SOMMERVILLE, 2006a).

- RF 1. Realizar una interfaz para modelar el tornillo y cargarlo a partir de los contenidos de una Base de datos (BD).
- RF 2. Insertar la opción para las diferentes vistas y herramientas.
- RF 3. Realizar una tabla que visualice los contenidos del tornillo reflejado en la BD.
- RF 4. Realizar árbol *dockeable* que muestre los distintos tipos de tornillos.
- RF 5. Mostrar los iconos de los tornillos.
- RF 6. Seleccionar un tornillo dentro del conjunto de tornillos.

- RF 7. Insertar una barra de herramienta que contenga opciones para la navegación.
- RF 8. Insertar botón para aceptar o cancelar la selección.
- RF 9. Dibujar con las funciones de Open Cascade el perfil del tornillo y convertirlo en un sólido, a partir de los contenidos de una BD.
- RF 10. Realizar una funcionalidad para aplicar textura con aspecto de rosca sobre la cara q contiene la rosca, a partir de los contenidos de una BD.
- RF 11. Visualizar el resultado en el visor de la aplicación a través del *ViewProvider*.
- RF 12. Insertar la opción de visualizar el tornillo modelado en el árbol de la aplicación.
- RF 13. Realizar la comunicación desde las interfaces de la aplicación.
- RF 14. Integrar el componente dentro del módulo ensamble de la aplicación.

### 2.3.2. Requisitos No Funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el procesos de desarrollo y estándares. Estos se clasifican en requisitos no funciones de usabilidad, de confiabilidad, de hardware, de software, de soporte, en restricciones de diseño e implementación, de eficiencia y de apariencia o interfaz externa. A continuación, se exponen los requisitos no funcionales del componente:

- RNF 1. Software: El sistema debe funcionar para el sistema operativo Linux.
- RNF 2. Restricciones de diseño e implementación: Arquitectura de llamada y retorno, patrón arquitectónico Modelo-Vista-Controlador, lenguaje de programación C++, biblioteca *OpenCascade* y *framework QtCreator*.
- RNF 3. Restricciones de diseño e implementación: Herramienta de modelado *Visual Paradigm* en su versión 8.0.
- RNF 4. Soporte: El sistema debe instalarse donde el cliente lo desee.

## 2.4. Historias de usuario

Una Historia de usuario (HU), es donde el cliente describe brevemente las características funcionales o no funcionales que el sistema debe poseer. El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento pueden eliminarse, reemplazarse por otras más específicas, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en poco tiempo(Jeffries, s.f.).

La HU presenta la siguiente estructura:

- **Número:** A cada HU se le asigna un número para facilitar su identificación.
- **Nombre:** Nombre descriptivo de la HU.
- **Prioridad:** Grado de prioridad que se le asigna a la HU en dependencia de las necesidades del cliente, (Alta, Media o Baja).
- **Complejidad:** Grado de complejidad que se le asigna a la HU luego de ser analizada. (Alta, Media o Baja).
- **Estimación:** Unidades de tiempo estimadas por el equipo de desarrollo para darle cumplimiento a la HU.
- **Iteración:** Número de la iteración en la cual será implementada la HU.
- **Descripción:** Descripción simple sobre lo que debe hacer la funcionalidad en cuestión.
- **Información Adicional:** Breve información que ayude a comprender algún dato no especificado antes.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
<b>Número:</b> 1	<b>Nombre:</b> Modelar el tornillo y cargarlo a partir de los contenidos de una (BD).
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite modelar el tornillo seleccionado a partir de los contenidos de una BD y mostrarlo en el visor de la aplicación.	
<b>Observaciones:</b> El tornillo debe estar seleccionado para poder modelarlo.	

Continúa en la próxima página

Tabla 2.1. Continuación de la página anterior

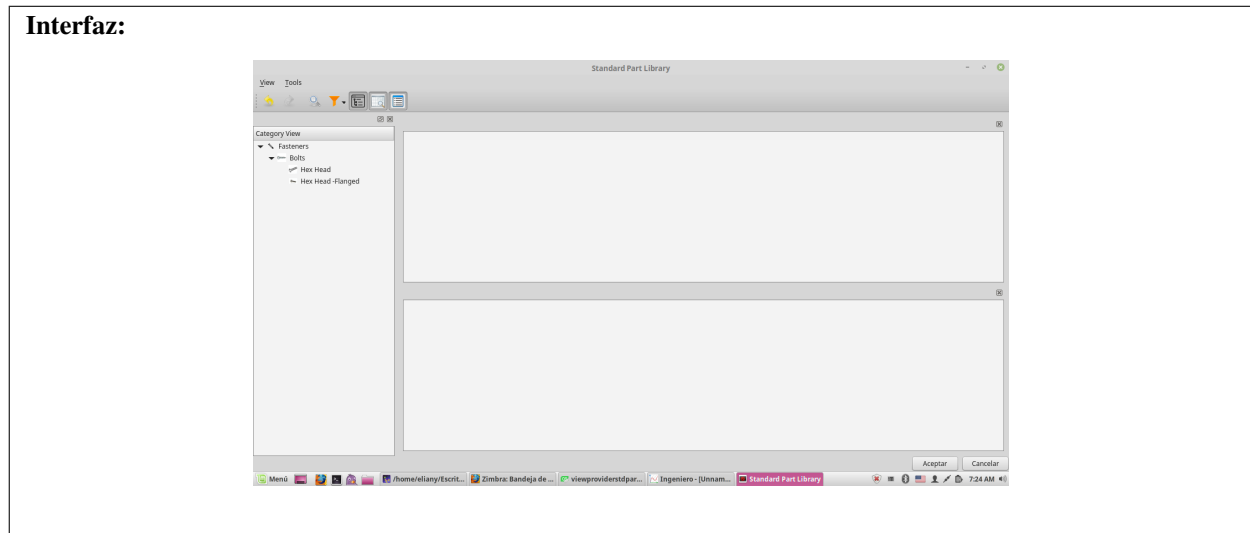


Tabla 2.2. Historia de usuario # 2

Historia de usuario	
<b>Número:</b> 2	<b>Nombre:</b> Insertar la opción para las diferentes vistas y herramientas.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Introducir los elementos necesarios que estarán contenidos en las opciones de Vista(View) y Herramientas(Tools).	
<b>Opción de Vista</b> Contendrá diferentes tipos de vistas como son en forma de árbol, de tabla y en forma de lista.	
<b>Opción de Herramientas</b> Contendrá la herramienta de búsqueda, de filtrado así como opciones para la navegación hacia delante y hacia atrás.	
<b>Observaciones:</b> Para realizar cualquiera de las opciones que ofrece la Vista y las Herramientas debe seleccionar al menos una.	

Continúa en la próxima página

Tabla 2.2. Continuación de la página anterior

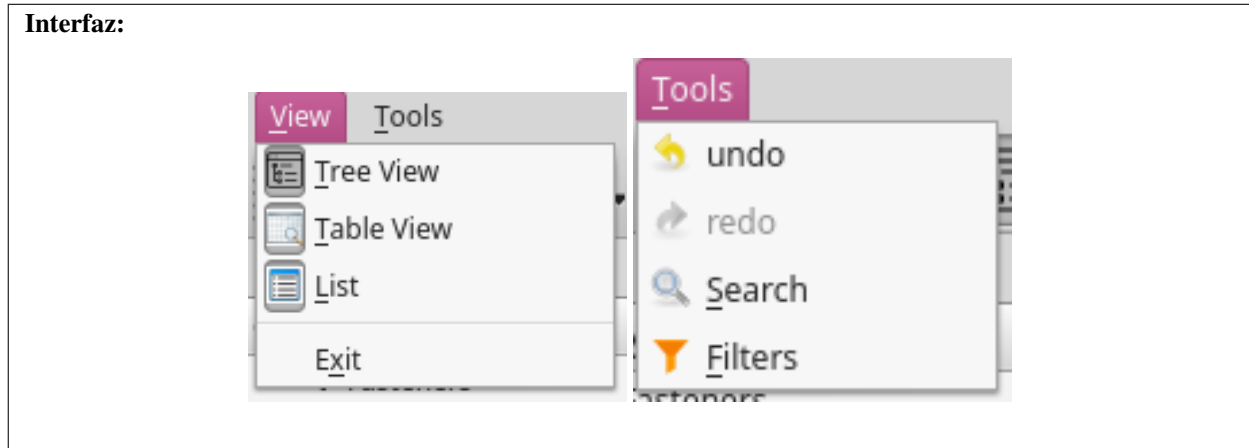


Tabla 2.3. Historia de usuario # 3

Historia de usuario	
<b>Número:</b> 3	<b>Nombre:</b> Realizar una tabla que visualice los contenidos del tornillo reflejado en la BD.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite mostrar una tabla que se encuentra en la BD con los datos del tornillo que se requiera modelar.	
<b>Observaciones:</b> Los datos de los tornillos ya deben estar contenidos en una BD.	
<b>Interfaz:</b>	

Tabla 2.4. Historia de usuario # 4

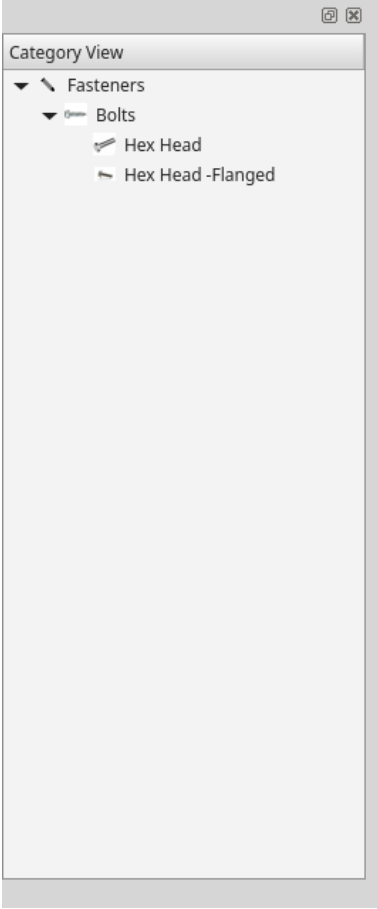
Historia de usuario	
<b>Número:</b> 4	<b>Nombre:</b> Realizar árbol <i>dockeable</i> que muestre los distintos tipos de tornillos.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 2
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite mostrar un árbol, el cual tiene agrupado los distintos tipos de tornillos. Este árbol se mostrará en cualquier sitio de la aplicación con tan solo hacer click por uno de sus extremos y arrastrarlo para el sitio deseado.	
<b>Observaciones:</b> El árbol se mostrará cuando el usuario marque la opción de <i>Tree View</i> contenida en el menú de la aplicación, específicamente en la Vista.	
<b>Interfaz:</b>	
	



Tabla 2.5. Historia de usuario # 5


Historia de usuario	
<b>Número:</b> 5	<b>Nombre:</b> Mostrar los iconos de los tornillos.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 2
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite mostrar en la parte izquierda de la aplicación los iconos de los distintos tipos de tornillos que puedan existir (ejemplo dentro de los Bolts, los del tipo Hex Head)	
<b>Observaciones:</b> Los iconos se mostrarán a menos que el usuario halla seleccionado el tipo de tornillo que desea diseñar.	
<b>Interfaz:</b>	
	

Tabla 2.6. Historia de usuario # 6

Historia de usuario	
<b>Número:</b> 6	<b>Nombre:</b> Seleccionar un tornillo dentro del conjunto de tornillos.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 2
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite seleccionar dentro de los distintos tipos de tornillos que puedan existir un tornillo.	
<b>Observaciones:</b> Se podrá seleccionar otro tipo de tornillo dentro del conjunto de tornillo, a menos que el usuario halla seleccionado el tipo de tornillo que quiera diseñar.	

Continúa en la próxima página

Tabla 2.6. Continuación de la página anterior

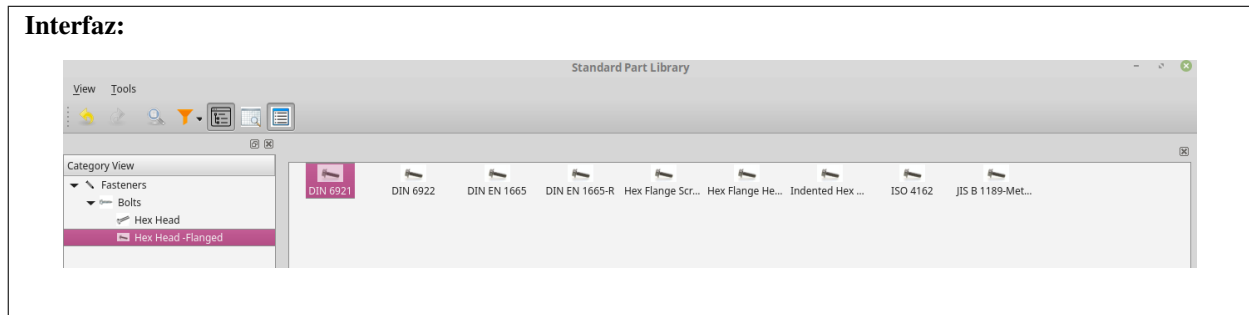


Tabla 2.7. Historia de usuario # 7

Historia de usuario	
<b>Número:</b> 7	<b>Nombre:</b> Insertar una barra de herramienta que contenga opciones para la navegación.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Muestra una serie de herramientas para un agilizado trabajo, mediante las cuales se podrá interactuar con la aplicación de forma más directa, sin tener la necesidad de desplegar la opción del menú dedicada a la parte de las herramientas.	
<b>Observaciones:</b> Se podrá seleccionar cualquiera de las herramientas a menos que la aplicación esté abierta.	
<b>Interfaz:</b>	

Tabla 2.8. Historia de usuario # 8

Historia de usuario	
<b>Número:</b> 8	<b>Nombre:</b> Insertar botón para aceptar o cancelar la selección.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	

Continúa en la próxima página

Tabla 2.8. Continuación de la página anterior


<p><b>Descripción:</b> Muestra dos botones en la parte inferior de la aplicación con las opciones de aceptar o cancelar la selección.</p> <p><b>Botón Aceptar</b> Cuando el usuario seleccione un tornillo para su modelado con sus respectivos datos, para visualizarlo debe marcar esta opción.</p> <p><b>Botón Cancelar</b> Cuando el usuario seleccione un tornillo para su modelado con sus respectivos datos, para visualizarlo debe dar la opción Aceptar, pero si su selección no fue la adecuada puede dar en esta opción de Cancelar.</p>
<p><b>Observaciones:</b> Se podrá seleccionar cualquiera de los botones si la aplicación está abierta y el usuario seleccionó el tornillo para su modelado.</p>
<p><b>Interfaz:</b></p> <div style="text-align: center;">  </div>

Tabla 2.9. Historia de usuario # 9

Historia de usuario	
<b>Número:</b> 9	<b>Nombre:</b> Dibujar con las funciones de OpenCascade el perfil del tornillo.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite a través de dichas funcionalidades de OpenCascade convertir el tornillo seleccionado en un sólido en dependencia de los datos contenidos en la Base de Datos.	
<b>Observaciones:</b> Se convertirá el tornillo en un sólido a menos que el usuario halla seleccionado los datos correspondientes.	

Continúa en la próxima página

Tabla 2.9. Continuación de la página anterior

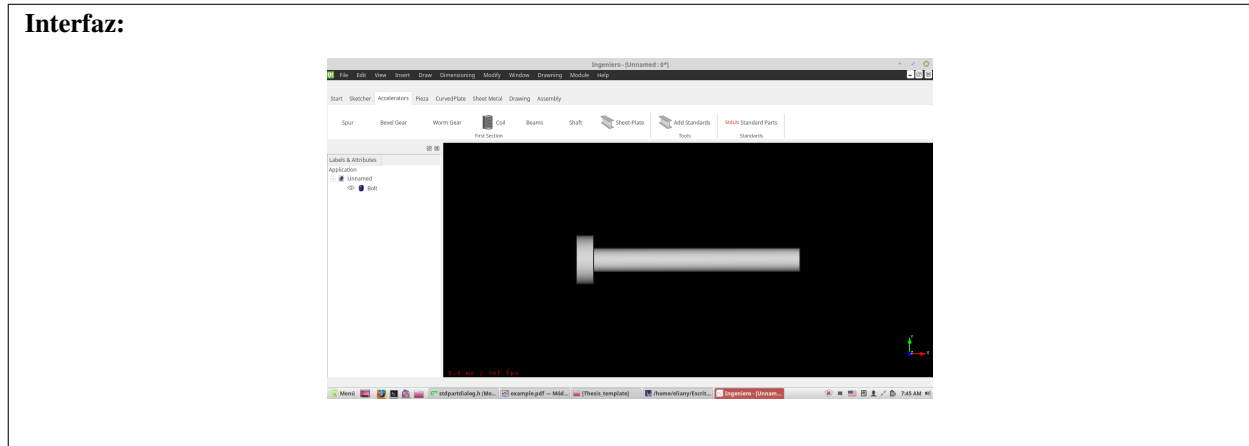


Tabla 2.10. Historia de usuario # 10

Historia de usuario	
<b>Número:</b> 10	<b>Nombre:</b> Realizar una funcionalidad para aplicar textura con aspecto de rosca.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite a través de las funcionalidades de OpenCascade convertir el tornillo seleccionado en un sólido en dependencia de los datos contenidos en al Base de Datos, así como aplicar textura con aspecto de rosca en la parte que contiene la rosca en el tornillo.	
<b>Observaciones:</b> Se realizará dicha funcionalidad cuando el usuario halla seleccionado el tornillo y a su vez la fila de la tabla correspondiente a dicho tornillo.	
<b>Interfaz:</b>	

Tabla 2.11. Historia de usuario # 11

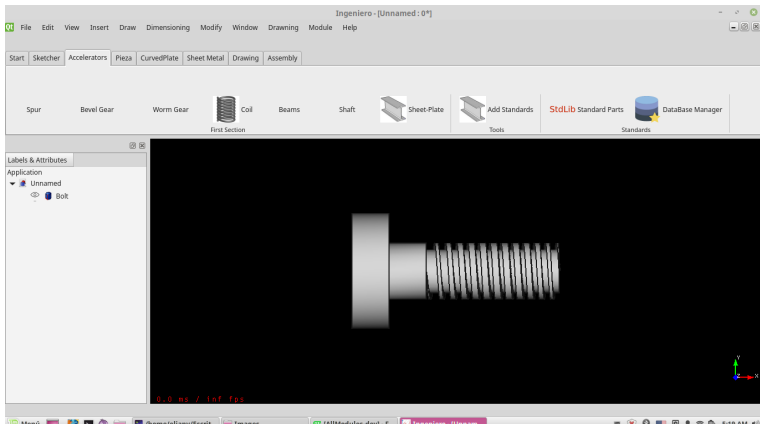
Historia de usuario	
<b>Número:</b> 11	<b>Nombre:</b> Visualizar el resultado en el visor de la aplicación a través del <i>ViewProvider</i> .
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite mostrar el tornillo seleccionado en el visor de la aplicación	
<b>Observaciones:</b> El tornillo se mostrará a menos que el usuario halla seleccionado el tornillo y la fila de la tabla con los datos correspondientes y le halla dado la opción de Aceptar.	
<b>Interfaz:</b>	
	

Tabla 2.12. Historia de usuario # 12

Historia de usuario	
<b>Número:</b> 12	<b>Nombre:</b> Insertar la opción de visualizar el tornillo modelado en el árbol de la aplicación.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 168h	<b>Iteración asignada:</b> 2
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite mostrar el tornillo seleccionado en el árbol de la aplicación una vez seleccionado y mostrado en el visor de la aplicación	
<b>Observaciones:</b> El tornillo se mostrará en el árbol a menos que el usuario ya halla seleccionado el tornillo y a su vez halla trabajado con él.	

Continúa en la próxima página

Tabla 2.12. Continuación de la página anterior

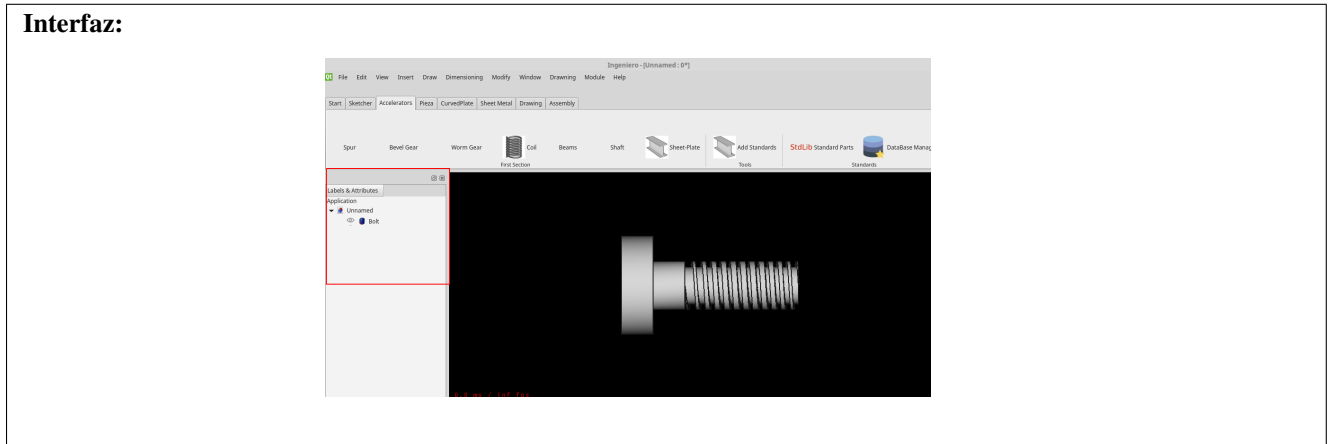


Tabla 2.13. Historia de usuario # 13

Historia de usuario	
<b>Número:</b> 13	<b>Nombre:</b> Realizar la comunicación desde las interfaces de la aplicación.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 1
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> La base de datos debe comunicarse con las interfaces de la aplicación mediante un servidor local.	
<b>Observaciones:</b> Para realizar la comunicación se debe tener en cuenta los siguientes datos: Host, Nombre de la base de datos, Nombre del usuario y Contraseña. El usuario introduce los datos para establecer la conexión y selecciona la opción “Test Connection”.	
Si se selecciona el botón Cancel se cierra la ventana.	
<b>Interfaz:</b>	

Tabla 2.14. Historia de usuario # 14

Historia de usuario	
<b>Número:</b> 14	<b>Nombre:</b> Integrar el componente dentro del módulo Ensamble de la aplicación.
<b>Usuario:</b>	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Bajo
<b>Puntos estimados:</b> 336h	<b>Iteración asignada:</b> 2
<b>Programador responsable:</b> Eliany Betancourt Lumpuy	
<b>Descripción:</b> Permite integrar el componente dentro del módulo de Ensamble de la aplicación para una mayor organización de la aplicación.	
<b>Observaciones:</b> El componente se integrará a menos que esté funcionando correctamente.	

## 2.5. Análisis y diseño

El diseño de software agrupa el conjunto de principios, conceptos y prácticas que llevan al desarrollo de un sistema o producto de alta calidad. Los principios de diseño establecen una filosofía general que guía el trabajo de diseño que debe ejecutarse. Deben entenderse los conceptos de diseño antes de aplicar la mecánica de éste, y la práctica del diseño en sí lleva a la creación de distintas representaciones del software que sirve como guía para la actividad de construcción que siga (Roger S. Pressman, [s.f.\[a\]](#)).

### 2.5.1. Estilo y patrón arquitectónico del software

Un estilo arquitectónico es una transformación que se impone al diseño de todo el sistema. El objetivo es establecer una estructura para todos los componentes del sistema. En el caso en el que ha de hacerse la reingeniería de una arquitectura ya existente la imposición de un estilo arquitectónico dará como resultado cambios fundamentales en la estructura del software, incluida la reasignación de las funciones de los componentes ([ibíd.](#)).

En cambio los patrones arquitectónicos se abocan a un problema de aplicación específica dentro de un contexto dado y sujeto a limitaciones y restricciones. El patrón propone una solución arquitectónica que sirve como base para el diseño de la arquitectura.

#### Estilo arquitectónico del software

La arquitectura de llamar y regresar. Este estilo arquitectónico permite obtener una estructura de programa que es relativamente fácil de modificar y escalar. Dentro de esta categoría existen varios sub\_estilos:

- Arquitecturas de programa principal/subprograma. Esta estructura clásica de programa descompone una función en una jerarquía de control en la que un programa “principal” invoca cierto número de componentes de programa que a su vez invocan a otros. La figura 2.1 ilustra una arquitectura de este tipo.

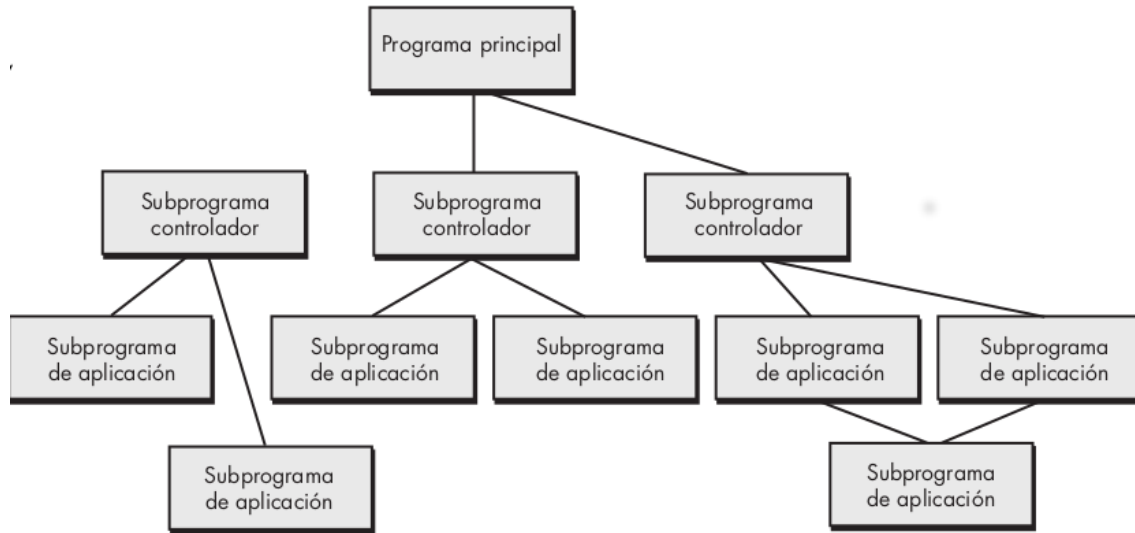


Figura 2.2. Arquitectura de programa principal/subprograma

- Arquitecturas de llamada de procedimiento remoto. Los componentes de una arquitectura de programa principal/subprograma están distribuidos a través de computadoras múltiples en una red.

El componente pertenece a un sistema que contiene un programa principal donde se invocan componentes para realizar las funcionalidades que necesita el usuario. Por tanto el estilo arquitectónico apropiado para el componente es llamada y retorno, en la categoría de arquitectura de programa principal/subprograma. Ver (Figura 2.2).

### Patrón arquitectónico del software

El patrón arquitectónico Modelo-Vista-Controlador, es el soporte del manejo de la interacción en muchos sistemas basados en la *Web*. Separa los componentes de la aplicación dependiendo de la responsabilidad que tienen, esto significa que cuando se hace un cambio en alguna parte del código, esto no afecte otra parte del mismo. Por ejemplo, si se modifica la Base de Datos, sólo se debería modificar el modelo que es quién se encarga de los datos y el resto de la aplicación debería permanecer intacta. Esto respeta el principio de la responsabilidad única (García, 2018).

- El Modelo que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
- La Vista, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste.
- El Controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.



### 2.5.2. Patrones de Diseño

Un patrón de diseño se caracteriza como “una regla de tres partes que expresa una relación entre cierto contexto, un problema y una solución”. Para el diseño de software, el contexto permite al lector entender el ambiente en el que reside el problema y qué solución sería apropiada en dicho ambiente. Un conjunto de requerimientos, incluidas limitaciones y restricciones, actúan como sistema de fuerzas que influyen en la manera en la que puede interpretarse el problema en este contexto y en cómo podría aplicarse con eficacia la solución(Larman, s.f.).

Los patrones *General Responsibility Assignment Software Patterns* (GRASP, por sus siglas en inglés) definidos en las funcionalidades:

- **Experto en Información:** Se asigna una responsabilidad al experto en información, la clase que tiene la información necesaria para realizar la responsabilidad. Este patrón se evidencia en la clase bolt que se encarga del modelado de los tornillos.
- **Creador:** Permite crear objetos de una clase determinada. Este patrón se evidencia en la clase de interfaz `stdpartdialog` ya que tiene la información necesaria para crear objetos del tipo bolt.
- **Polimorfismo:** se emplea para asignar comportamientos distintos según el tipo de clase. Este patrón se evidencia en la clase bolt que heredará de la clase Feature y reimplementa los métodos `execute` y `mustExecute`.
- **Alta Cohesión:** Se encarga de asignar una responsabilidad de modo que la cohesión siga siendo alta. Es la forma de medir cuán relacionadas están las responsabilidades de una clase. Una clase con alta cohesión es fácil de entender, reutilizar y mantener. Este patrón se evidencia en la clase *configurationdatabase* ya que pueden trabajar con la información de la base de datos sin una gran cantidad de trabajo.

**Patrón *Gang-of-Four* (GOF, por sus siglas en inglés o “pandilla de los cuatro”) definido en las funcionalidades:**

El patrón que se evidencia es el Observador en la funcionalidad `mustExecute`, dicho patrón define una dependencia del tipo de uno a muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los que dependen de él.

## 2.6. Diagrama de clases del diseño

Un Diagrama de Clases del Diseño muestra la especificación para las clases de una aplicación. Este incluye clases, asociaciones, atributos, interfaces, con sus operaciones y constantes, métodos y dependencias, ver (Figura 2.3). El componente cuneta con 6 clases y 9 funcionalidades.

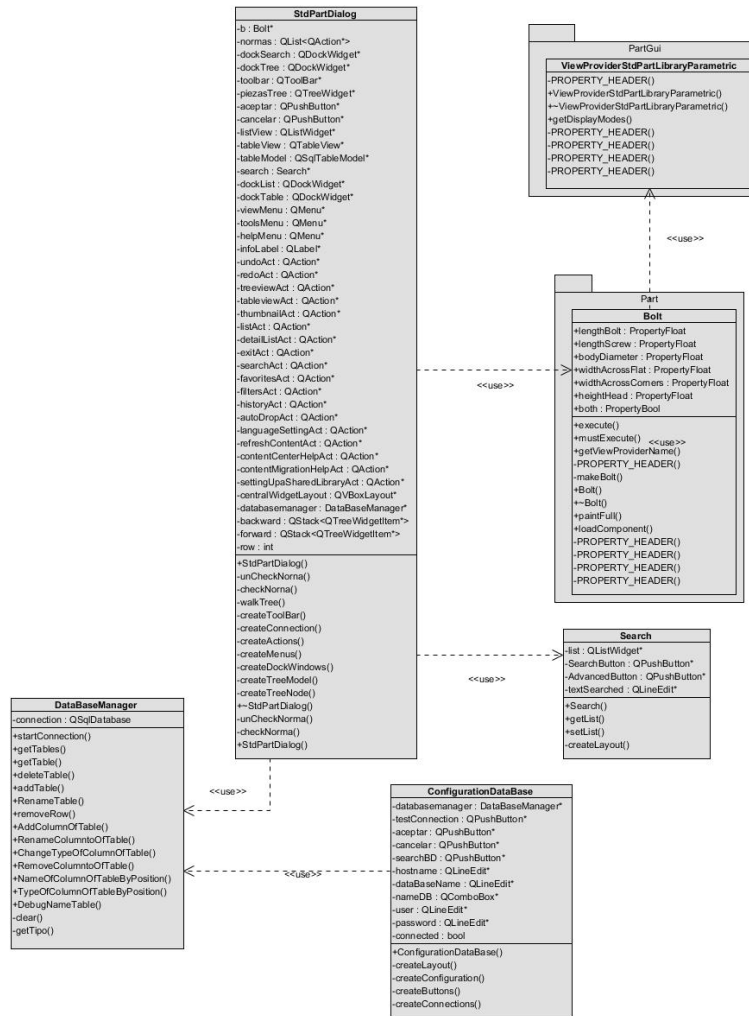


Figura 2.3. Diagrama de clases del diseño

- *StdPartDialog* (interfaz visual).
- *Search* (encargada de realizar las búsqueda respecto a las normas existentes).
- *Bolt* (encargada de contener los elementos necesarios para el modelado del tornillo).
- *ViewProviderStdPartLibraryParametric* (encargada de contener los elementos necesarios para visualizar el tornillo en el visor de la aplicación “Ingeniero”).
- *ConfigurationDataBase* (encargada de establecer la conexión con la base de datos).

En dicho diagrama se incluye la clase *DataBaseManager* ya que muchas de sus funcionalidades son utilizadas por varias de las clases descritas anteriormente.

## **2.7. Conclusiones del Capítulo**

- Después de obtener el resultado de la propuesta de solución, se evidenció que el componente cumple con todos los requisitos funcionales y no funcionales expuestos anteriormente.
- Las Historias de usuario permitieron dar una información detallada de cada uno de los requisitos funcionales para lograr un mejor entendimiento por parte del cliente.
- El diagrama de clases ayudó a una mejor comprensión de la estructura del componente, identificando así sus clases y funcionalidades.

---

## Implementación y pruebas

---

En el presente capítulo se abordan cada uno de los componentes que integran la etapa de implementación y prueba del componente a desarrollar. Se exponen los diseños de los casos de pruebas y el estándar de codificación, así como los resultados obtenidos del proceso de verificación de la calidad.

### 3.1. Implementación

La etapa de implementación del *software* es el proceso de convertir una especificación del sistema en un sistema ejecutable. Siempre implica los procesos de diseño y programación del *software*. Además incluye como entradas los artefactos de la etapa de diseño descritos anteriormente como diagramas de clases, especificación de la arquitectura, patrones a emplear en el sistema entre otros. El componente presenta una *Application Programming Interface*, que permite al usuario interactuar con los distintos tipos de tornillos existentes así como las normas asociadas a cada uno, las cuales están contempladas en una Base de datos. El componente cuenta con un total de 5 archivos fuentes (.cpp) y 6 archivos cabeceras (.h), un total de 9 funcionalidades, 1582 líneas de código y un total de 386 líneas en blanco.

#### 3.1.1. Detalles técnicos de la implementación

En la implementación del componente se utilizaron diferentes *Application Programming Interface* de la biblioteca de modelado *OpenCascade* como:

- *BRepBuilderAPI\_Transform*, esta API se utilizó para rotar modelos en la construcción de figuras para lograr el perfil del tornillo.
- *BRepBuilderAPI\_MakeFace*, con esta API se construyeron superficies simples, como las caras de las figuras que conformarían el perfil del tornillo.

Descripción	Ejemplo
<b>Definición de Objetos, Clases, funciones y atributos</b>	
Todos los nombres de las clases implementadas comenzarán con letra mayúscula. En caso de poseer un nombre compuesto se escribirán de acuerdo a la normativa CamelCase-UpperCamelCase.	<pre>class Foo{   cuerpo de la clase } class FooFirst{   cuerpo de la clase }</pre>
Siempre se declara para todas las clases implementadas su respectivo destructor de clase.	<pre>virtual ~Foo()</pre>
La declaración de funciones o métodos siempre comenzarán en letra inicial minúscula. En caso de ser un nombre compuesto se registrará por la normativa CamelCase-lowerCamelCase.	<pre>&lt;Tipo dato retorno&gt; funcion() &lt;Tipo dato retorno&gt; funcionCompuesta() &lt;Tipo dato retorno&gt;funcionDobleCompuesta()</pre>

Figura 3.1. Estándar de codificación

### 3.1.2. Estándar de codificación

Los estándares de codificación son pautas a cumplir por el código realizado en determinado sistema. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico, con el objetivo de garantizar que todos los participantes lo puedan entender en el menor tiempo y que el código en consecuencia sea fácil de modificar. (ver Figuras 3.1, 3.2, 3.3).

Un ejemplo de la normativa *CamelCase* se evidencia en el código de la clase `StdPartDialog`. (ver Figura 3.4).

### 3.1.3. Resultados de la implementación

En esta sección se muestran imágenes sobre el resultado de la implementación de las funcionalidades. En las figuras 3.5, 3.6, 3.7, 3.8 y 3.9 se muestra el resultado de las funcionalidades de las herramientas de filtrado, de búsqueda y de las vistas en forma de árbol, de tabla y de lista así como las opciones de navegación hacia delante y hacia atrás y la configuración con la Base de datos. También se muestran los resultados de un modelado de un tornillo.

## 3.2. Pruebas

El objetivo de la etapa de pruebas es descubrir defectos probando componentes de programas individuales. Estos componentes pueden ser funciones, objetos, o componentes reutilizables. Este proceso de pruebas

Los atributos siempre estarán escritos con letra minúscula. En caso de ser un nombre compuesto se registrará por la normativa CamelCase-lowerCamelCase.	<Tipo dato> atributo; <Tipo dato> atributoNombreCompuesto;
<b>Definición de parámetros dentro de las funciones y constructores de clases</b>	
Los nombres de los identificadores de los parámetros en las funciones deben estar escritos con minúscula separados a un espacio cada uno y en caso de ser compuesto utilizar normativa CamelCase-lowerCamelCase.	<Tipo dato retorno> funcion(<tipo><id1>, <tipo><id2>, <tipo><idN>)
Los identificadores de los parámetros dentro de los constructores de las clases deben estar escritos con minúscula separados a un espacio cada uno y en caso de ser compuesto utilizar normativa CamelCase-lowerCamelCase.	Clase(<tipo><id1>, <tipo><id2>, <tipo><idN>)
<b>Definición de expresiones</b>	
Para una mejor comprensión en la lectura y legibilidad del código los operadores binarios exceptuando los punteros, función de llamado a miembros, escritura de un arreglo y paréntesis de una función se escribirán con un espacio entre ellos	x + y; x == y; idFuncion.miembro(); idFuncion->miembro(); array[];

Figura 3.2. Estándar de codificación

<b>Definición de estructuras de control y bucles</b>	
Las estructuras de control y los bucles estarán definidos de igual manera en ambos casos siguiendo el estándar determinado por el <i>framework</i> de Qt.	Para las estructuras if, else, if else : <estructura control>(condición){ tarea a ejecutar } Para los bucles while, for, do while y otros: <bucle>(condiciones){ tarea a ejecutar }
<b>Comentarios en el código según el estándar de C++.</b>	
Comentarios pequeños.	/* comentario sencillo */
Otros comentarios.	/* *Comentario */
Comentario de versión, descripción de clase y otras características de la clase o paquete.	/* ***** *Comentario amplio * ***** */

Figura 3.3. Estándar de codificación

```
void StdPartDialog::createMenus()  
{  
    viewMenu = menuBar()->addMenu(tr("&View"));  
    viewMenu->addAction(treeviewAct);  
    viewMenu->addAction(tableviewAct);  
    viewMenu->addAction(listAct);  
    viewMenu->addSeparator();  
    viewMenu->addAction(exitAct);  
  
    toolsMenu = menuBar()->addMenu(tr("&Tools"));  
    toolsMenu->addAction(undoAct);  
    toolsMenu->addAction(redoAct);  
    toolsMenu->addAction(searchAct);  
    toolsMenu->addAction(filtersAct);  
    toolsMenu->addSeparator();  
  
}
```

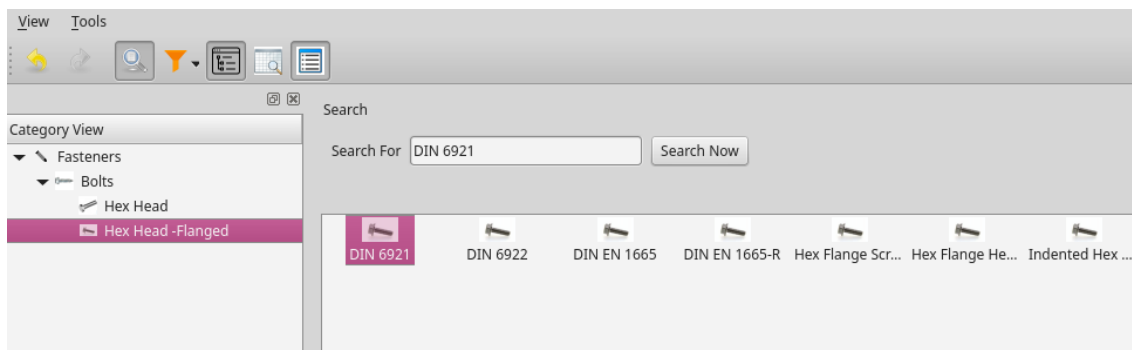
Figura 3.4. Ejemplo de la normativa *CamelCase*

Figura 3.5. Funcionalidad buscar

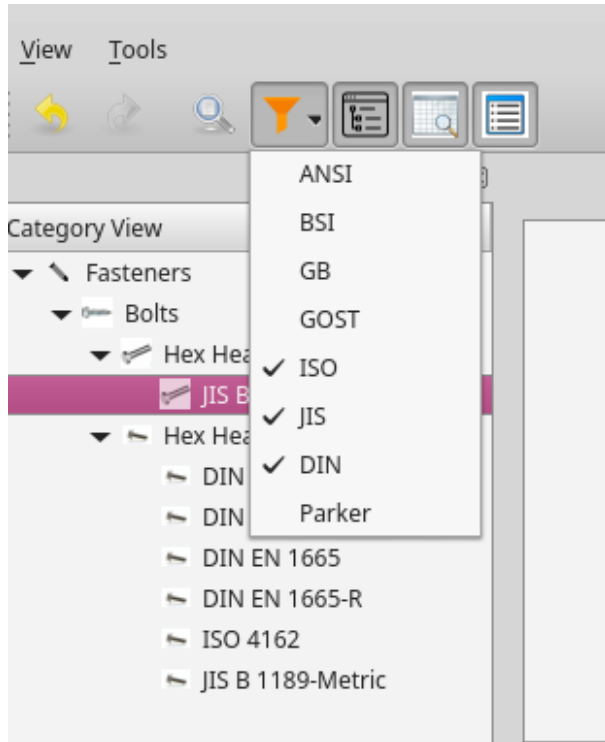


Figura 3.6. Funcionalidad filtrar

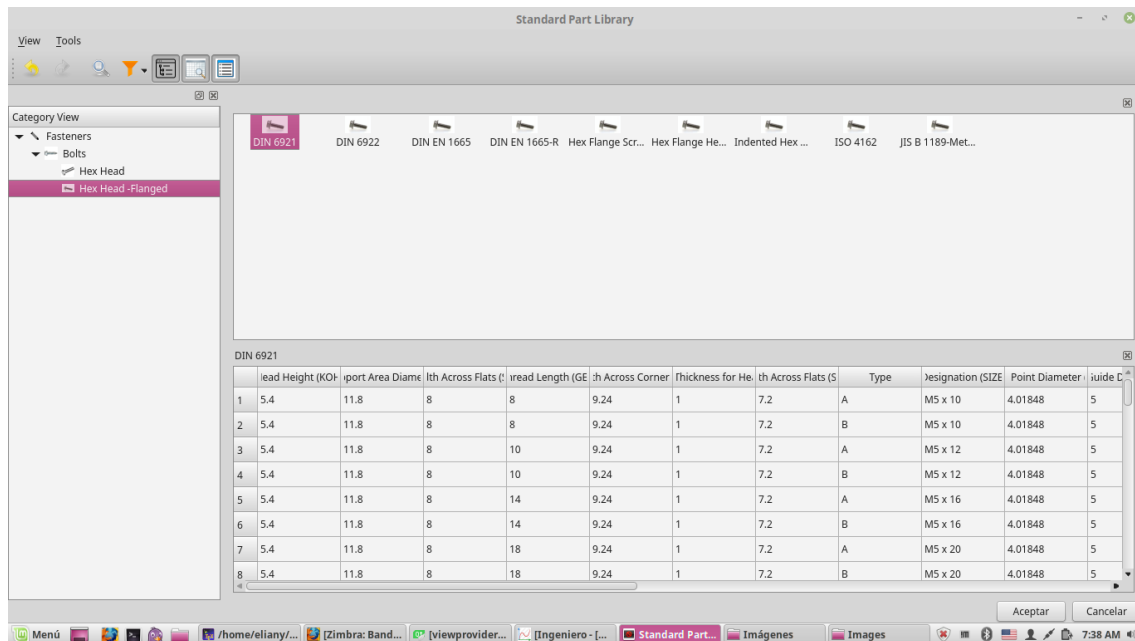


Figura 3.7. Vista en forma de tabla



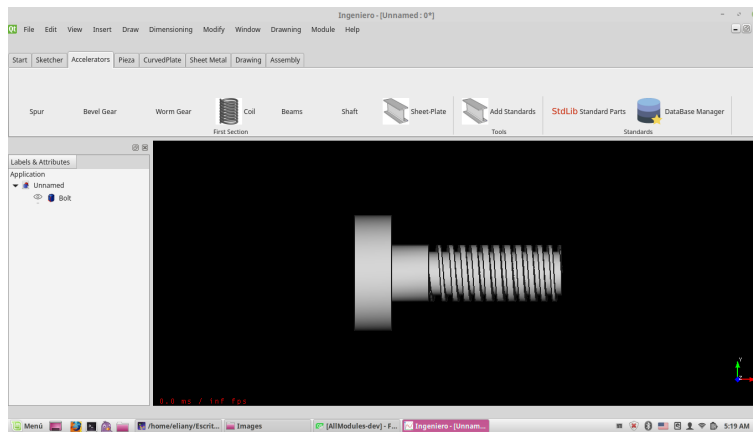


Figura 3.8. Diseño de un tornillo

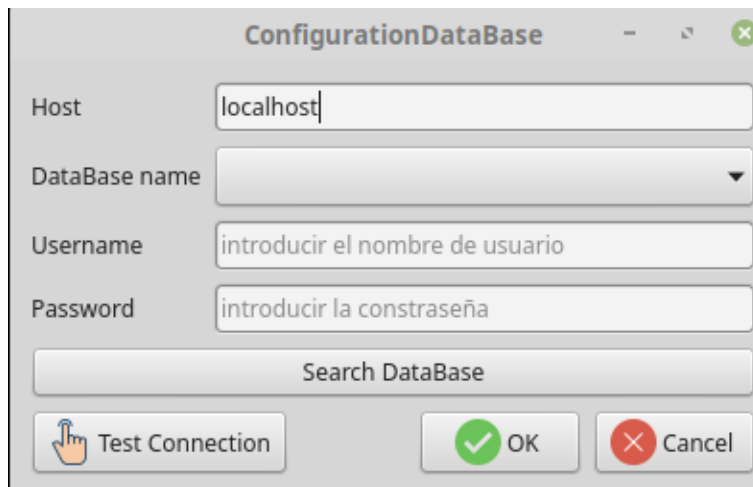


Figura 3.9. Configuración para la conexión con la Base de datos

tiene dos metas distintas(SOMMERVILLE, 2006b):

- Demostrar al desarrollador y al cliente que el software satisface sus requisitos. Esto significa que debería haber al menos una prueba para cada requerimiento o característica que se incorporará a la entrega del producto.
- Descubrir defectos en el software en el que el comportamiento de este es incorrecto, no deseable o no cumple su especificación. La prueba de defectos está relacionada con la eliminación de todos los tipos de comportamientos del sistema no deseables, tales como caídas del sistema, interacciones no permitidas con otros sistemas, cálculos incorrectos y corrupción de datos.

### 3.2.1. Niveles de prueba

Durante el desarrollo, las pruebas se realizan en tres niveles de granulación(Sommerville, s.f.):

- Pruebas de unidad o pruebas unitarias, donde se ponen a prueba unidades de programa o clases de objetos individuales. Las pruebas de unidad deben enfocarse en comprobar la funcionalidad de objetos o métodos.
- Pruebas del componente o pruebas de integración, donde muchas unidades individuales se integran para crear componentes compuestos. La prueba de componentes debe enfocarse en probar interfaces del componente.
- Pruebas del sistema, donde algunos o todos los componentes en un sistema se integran y el sistema se prueba como un todo. Las pruebas del sistema deben enfocarse en poner a prueba las interacciones de los componentes.
- Pruebas de aceptación, donde los clientes prueban un sistema para decidir si está o no listo para ser aceptado por los desarrolladores del sistema y desplegado en el entorno del cliente.

Las pruebas de desarrollo son, ante todo, un proceso de prueba de defecto, en las cuales la meta consiste en descubrir bugs en el software. Por lo tanto, a menudo están entrelazadas con la depuración: el proceso de localizar problemas con el código y cambiar el programa para corregirlos.

### 3.2.2. Métodos de prueba

Con el objetivo de encontrar los defectos del software se diseñan los casos de pruebas, mediante los cuales se obtienen un conjunto de pruebas, para lograr este objetivo existen dos métodos de prueba:

- Prueba de Caja Blanca: se basa en el examen cercano de los detalles de procedimiento. Las rutas lógicas a través del software y las colaboraciones entre componentes se ponen a prueba al revisar conjuntos específicos de condiciones y/o bucles(Roger S. Pressman, s.f.[b]).

- Prueba de Caja Negra: se refiere a las pruebas que se llevan a cabo en la interfaz del software. Una prueba de caja negra examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del software (Roger S. Pressman, s.f.[b]).

### 3.2.3. Diseño de Casos de Prueba

Los casos de prueba son la descripción de las actividades que se van a ejecutar con el fin de validar la aplicación, así como incluyen toda las funciones que el programa sea capaz de hacer. Poseen una serie de características como:

- Se prueba el programa completo.
- Uno o varios casos de prueba por cada requisito.
- Se usan las mismas técnicas, pero con otro objetivo.

Los Casos de Pruebas, como una técnica de partición y equivalencia, han sido realizados sobre la base de las Historias de Usuarios y tienen como objetivo fundamental encontrar la mayor cantidad posible de deficiencias existentes en las funcionalidades implementadas, (ver Tabla 3.1 y 3.2 y Anexo A.1).

Tabla 3.1. Diseño de Casos de Prueba RF 1

Descripción general			
Permite modelar el tornillo seleccionado a partir de las normas existentes.			
SC 1 Seleccionar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción View.	Selecciona la opción View la cual mostrará una serie de opciones para visualizar los tornillos en forma de tabla, de árbol y de lista.	Brinda la posibilidad de seleccionar las distintas formas de vista de los tornillos.	Accelerators /Standard Parts /View
EC 1.2 Opción de TreeView	Selecciona la opción de TreeView	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee.	Accelerators /Standard Parts /View /TreeView
SC 2 Modelar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Opción TableView.	Selecciona la opción TableView.	Brinda la posibilidad de ver los datos de los tornillos en forma de tabla y seleccionar una fila para modelar el tornillo que desee.	Accelerators /Standard Parts /View /TreeView
EC 2.2 Opción de Aceptar	Selecciona la opción de Aceptar	Muestra el diseño del tornillo seleccionado en el visor de la aplicación y cierra la ventana.	Accelerators /Standard Parts /Aceptar

EC 2.3 Opción de Cancelar	Selecciona la opción de Cancelar	Cierra la ventana.	Accelerators /Standard Parts /Cancelar
---------------------------	----------------------------------	--------------------	--

Tabla 3.2. Diseño de Casos de Prueba RF 9

Descripción general			
Permite convertir el tornillo seleccionado en un sólido.			
SC 1 Seleccionar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción View.	Selecciona la opción View la cual mostrará una serie de opciones para visualizar los tornillos en forma de tabla, de árbol y de lista.	Brinda la posibilidad de seleccionar las distintas formas de vista de los tornillos.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View
EC 1.2 Opción de TreeView	Selecciona la opción de TreeView	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView
SC 2 Modelar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Opción TreeView.	Selecciona la opción TreeView.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView
EC 2.2 Opción de Aceptar	Selecciona la opción de Aceptar	Muestra el diseño del tornillo seleccionado en el visor de la aplicación y cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Aceptar
EC 2.3 Opción de Cancelar	Selecciona la opción de Cancelar	Cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Cancelar

### 3.2.4. Resultados obtenidos de las pruebas funcionales

El resultado de las pruebas se tiene en cuenta para obtener las fallas que presenta el sistema y poder analizar las futuras, corregirlas permite disminuir el número de no conformidades posibles, mejorando el grado de aceptación por parte del cliente.

La realización de las pruebas permitió identificar una serie de no conformidades que se pueden observar en la tabla 3.3:

Tabla 3.3. No Conformidades

No. NC	Requisito Funcional	Descripción	Complejidad	Estado
1	RF 1	No mostraba los tornillos en el árbol.	Baja	Resuelta
2	RF 1	No se pueden hacer varias instancias del mismo tornillo en el visor de la aplicación.	Alta	No resuelta
3	RF 2	No mostraba la vista en forma de lista.	Baja	Resuelta
4	RF 2	No cerraba la ventana en la opción exit.	Baja	Resuelta
5	RF 2	No mostraba la opción de buscar.	Baja	Resuelta
6	RF 3	Mostraba la tabla vacía	Alta	Resuelta
7	RF 3	No mostraba la tabla del tornillo seleccionado.	Alta	Resuelta
8	RF 4	No se mostraba el árbol a la derecha de la ventana.	Baja	Resuelta
9	RF 6	No se seleccionaba un tornillo en específico.	Alta	Resuelta
10	RF 7	No filtraba por las normas.	Alta	Resuelta
11	RF 7	No funcionaba la herramienta de buscar	Alta	Resuelta
12	RF 8	Se cerraba la ventana <i>Std Part Library</i> , al presionar el botón Aceptar sin seleccionar una fila de la tabla.	Baja	No resuelta

Se puede afirmar que la efectividad del componente para el modelado de tornillos, está avalada por los resultados de las pruebas realizadas por el equipo de desarrollo y especialistas en el diseño de piezas mecánicas.

En la figura 3.10 se muestran los datos correspondientes a cada iteración de prueba por las que transitó el componente.

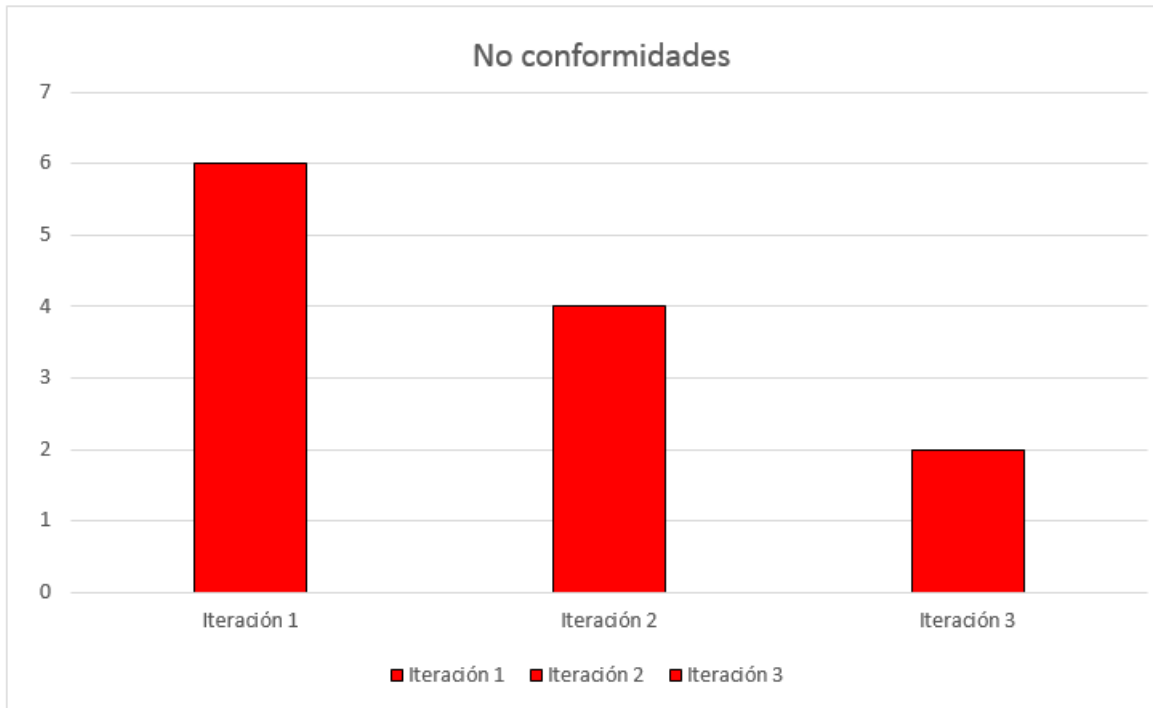


Figura 3.10. Iteraciones de las pruebas

### 3.2.5. Resultados obtenidos de las pruebas de integración

Las pruebas de integración comprueban que los componentes realmente funcionan juntos. Para la realización de esta prueba se tuvo en cuenta la estrategia Big-Bang, la cual es la única estrategia que no es incremental y se utiliza únicamente cuando se tiene todos los componentes a integrar.

### 3.2.6. Pruebas de Aceptación

Ésta es la etapa final en el proceso de pruebas, antes de que el sistema se acepte para uso operacional. El sistema se pone a prueba con datos suministrados por el cliente del sistema, en vez de datos de prueba simulados. Las pruebas de aceptación revelan los errores y las omisiones en la definición de requerimientos del sistema, ya que los datos reales ejercitan el sistema en diferentes formas a partir de los datos de prueba (Sommerville, s.f.).

A los estudiantes y profesores, pertenecientes al grupo de investigación Soluciones Informáticas para la Ingeniería y la Industria, se les realizó una encuesta de opinión el día 6 de mayo a las 2:00 de la tarde. Los criterios acerca del componente no fueron en su totalidad positivos, lo que arrojó un 80% de aceptación por parte de los encuestados.

### 3.3. Conclusiones del Capítulo

- Con el propósito de validar las funcionalidades desarrolladas se realizaron pruebas al software, las cuales detectaron 12 no conformidades de las cuales 2 no fueron corregidas, lo que determina que la calidad del componente no está valorada para un 100%.
- Se definieron los estándares de codificación establecidos por el grupo de investigación para la implementación de las clases y los métodos definidos en la fase de diseño.

Como resultado del proceso de Investigación y Desarrollo realizado se arribó a las siguientes conclusiones generales:

- Mediante la conexión de la aplicación con la base de datos de componentes normalizados, se logró obtener y visualizar los datos de cada tornillo según las distintas normas existentes, en forma de tabla.
- Con la implementación de las funcionalidades del componente, se logró acelerar el proceso de modelado de tornillos, a partir de una norma determinada, disminuyendo así el tiempo de trabajo de los diseñadores, ingenieros mecánicos entre otros.
- El resultado obtenido refleja un componente que puede ser adaptado a cualquier aplicación de código abierto que emplee tecnologías de OpenCascade.



---

## Recomendaciones

---

A partir del estudio realizado y luego de haber analizado los resultados obtenidos se recomiendan los siguientes elementos a tener en cuenta para futuros trabajos:

- Completar la funcionalidad de modelar los componentes normalizados para otro tipo de elemento de fijación, es decir, para tuercas, pasadores, remaches y arandelas.
- Completar la funcionalidad de modelar tornillos según la norma necesaria.

---

## Referencias bibliográficas

---

- A.ENRIQUEZ.HZ. El tornillo: un invento que sostiene al mundo. Url: <http://e-Historia.es> (vid. pág. 6).
- ADINA-M, 2015. Url: <http://www.adina.com/adinam.shtml> (vid. pág. 2).
- ANGELES VIRGUEZ, María de los, 2015. *Los 36 Tipos de Tornillos Principales y sus Usos*. Url: <https://www.lifeder.xn--com-tecnologa-7ib> (vid. pág. 7).
- Áreas Técnicas. Url: <http://www.unizar.es/aeipro/finder/INGENIERIA%20DE%20PRODUCTOS/BF04..htm> (vid. pág. 5).
- Autodesk Inventor Professional 2017. 2018. Url: <http://www.imaginit.com/software/autodesk-products/inventor> (vid. págs. 17, 19).
- BÚA, Manuel Torres, 2014. *Normalización*. Url: [https://www.edu.xunta.es/espazoAbalar/sites/espazoAbalar/files/datos/1464946300/contido/4\\_normalizacin.html](https://www.edu.xunta.es/espazoAbalar/sites/espazoAbalar/files/datos/1464946300/contido/4_normalizacin.html) (vid. pág. 15).
- CASANOVA, Félix. *Inventos: El tornillo*. Url: <https://hdnh.es/inventos-el-tornillo/> (vid. pág. 14).
- CEJAROSU, 2006. *Tornillo*. Url: <http://concurso.cnice.mec.es/cnice2006/material107/index.htm> (vid. págs. 6, 7, 9, 11, 13).
- CIENCIAS INFORMÁTICAS, Universidad de las (ed.). *Metodología de desarrollo para la Actividad productiva de la UCI.pdf* (vid. pág. 21).
- Dtecnico, 2018. Url: [http://www.vc.ehu.es/Dtecnico/tema12\\_05.htm](http://www.vc.ehu.es/Dtecnico/tema12_05.htm) (vid. pág. 12).
- EDEKI, Charles. *Agile Unified Process*. N.º 1. Url: <http://www.ijcsma.com/publications/september2013/V1I304.pdf> (vid. pág. 21).
- Enciclopedia de ciencia y técnica: Tornillos y tuercas*, 1984. ISBN ISBN 84-345-4490-3 (vid. pág. 2).
- GARCÍA, Miriam, 2018. *MVC (Modelo-Vista-Controlador): ¿qué es y para qué sirve?* Url: <https://codingornot.com/mvc-modelo-vista-controlador-que-es-y-para-que-sirve> (vid. pág. 37).
- JEFFRIES R., A. Anderson y C. Hendrickson. *Extreme Programming Installed* (vid. pág. 25).
- LARMAN, Craig. *UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado*. 2da (vid. pág. 38).

- Las normas más habituales en tornillería*, 2017. Url: <https://www.tormetal.com> (vid. pág. 16).
- OCA MONTANO, José Luis Montes de. La migración hacia software libre en Cuba: complejo conjunto de factores sociales y tecnológicos en el camino de la soberanía nacional. N.º 7, págs. p.119\_125.. Url: <http://rus.ucf.edu.cu/> (vid. págs. 1, 2).
- Open CASCADE Technology*, 2018. Url: [http://dev.opencascade.org/doc/overview/html/index.html#OCCT\\_OVW\\_SECTION\\_1](http://dev.opencascade.org/doc/overview/html/index.html#OCCT_OVW_SECTION_1) (vid. pág. 23).
- OSWALDO ROJAS LAZO, Luis Rojas Rojas, 2018. *DISEÑO Y TECNOLOGÍA*. Url: [http://sisbib.unmsm.edu.pe/bibvirtualdata/publicaciones/indata/vol9\\_n1/a02.pdf](http://sisbib.unmsm.edu.pe/bibvirtualdata/publicaciones/indata/vol9_n1/a02.pdf) (vid. pág. 1).
- ROGER S. PRESSMAN, Ph.D., (a). *Ingeniería del software UN ENFOQUE PRÁCTICO* (vid. pág. 36).
- ROGER S. PRESSMAN, Ph.D., (b). *Ingeniería del software, UN ENFOQUE PRÁCTICO*. SÉPTIMA EDICIÓN (vid. págs. 47, 48).
- Solid Edge V15 añade biblioteca de Standard Parts*, 2018. Url: <http://www.xn--solid%20edge%20v15%20aade%20biblioteca%20de%20standard%20parts%20-5ve?%20Pixel%20Sistemas>. (vid. pág. 20).
- Solidworks Company History*, 2018. Url: <http://www.solidworks.com> (vid. pág. 19).
- SOMMERVILLE, IAN, 2006a. *Ingeniería de Software*. 8va (vid. pág. 24).
- SOMMERVILLE, IAN, 2006b. *Ingeniería de Software*. 7ma (vid. pág. 47).
- SOMMERVILLE, Ian. *INGENIERÍA DE SOFTWARE*. Novena edición (vid. págs. 47, 51).
- Spatial Releases 2017 1.0, Delivering Technology Enhancements Aimed at Innovation and Industrialization*, 2016. Url: <https://www.spatial.com/news/spatial-releases-2017-10-delivering-technology-enhancements-aimed-innovation-and> (vid. pág. 1).

# **Apéndices**

### A.1. Casos de prueba

Tabla A.1. Diseño de Casos de Prueba RF №2

Descripción general				
Permite interactuar con las opciones de Vista y Herramienta del menú.				
SC 1 Opción de las Vistas.				
Escenario	Descripción	Respuesta del sistema	Flujo central	
EC 1.1 Opción TreeView.	Selecciona la opción TreeView la cual mostrará una vista en forma de árbol.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary /View /TreeView	
EC 1.2 Opción de TableView.	Selecciona la opción TableView.	Brinda la posibilidad de visualizar los datos de los tornillos en forma de tabla y seleccionar una fila para modelar el tornillo que desee.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary /View /TableView	

EC 1.3 Opción List.	Selecciona la opción List.	Brinda la posibilidad de visualizar los tornillos en forma de lista.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary /View /List	
EC 1.4 Opción Exit	Selecciona la opción Exit.	Cierra la ventana.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary /View /Exit	
SC 2 Opción de las Herramientas				
Escenario	Descripción	Search For	Respuesta del sistema	Flujo central
EC 2.1 Opción Undo.	Selecciona la opción Undo la cual posibilita regresar al paso anterior de la selección.		Brinda la posibilidad de retornar al paso anterior a menos que hallas hecho una selección.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary  /Tools /Undo
EC 2.2 Opción Redo	Selecciona la opción Redo la cual posibilita ir hacia el siguiente paso.		Brinda la posibilidad de realizar la misma selección, una vez hallas hecho el undo.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Tools /Redo
EC 2.3 Opción Search	Selecciona la opción Search.	DIN 6921	Brinda la posibilidad de buscar los tornillos por su nombre.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Tools /Search
EC 2.4 Opción Filters	Selecciona la opción Filters.		Brinda la posibilidad de filtrar por distintas normas a la vez.	Accelerators /Standard Parts /ConfigurationDataBase /StdPartLibrary /Tools /Filters

Tabla A.2. Diseño de Casos de Prueba RF №3

Descripción general			
Permite obtener una tabla con los datos de los tornillos.			
SC 1 Obtener tabla.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción TreeView.	Selecciona la opción TreeView la cual mostrará una vista en forma de árbol.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView.
EC 1.2 Opción de TableView	Selecciona la opción TableView.	Brinda la posibilidad de visualizar los datos del tornillo en forma de tabla y seleccionar una fila para modelar el tornillo que desee.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TableView.

Tabla A.3. Diseño de Casos de Prueba RF №4

Descripción general			
Permite mover el árbol que contiene los tornillos hacia cualquier lugar de la interfaz.			
SC 1 Abrir la interfaz.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción Standard Parts.	Selecciona la opción Standard Parts, la cual mostrará una ventana que tendrá los distintos campos para la conexión con la BD.	Brinda la posibilidad de establecer la conexión con la BD.	Accelerators /Standard Parts /ConfigurationDataBase
EC 1.2 Opción de TreeView	Selecciona la opción TreeView la cual mostrará una vista en forma de árbol.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView

Tabla A.4. Diseño de Casos de Prueba RF №5

Descripción general
Permite visualizar los iconos de los tornillos.

SC 1 Mostrar iconos de los tornillos.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción TreeView.	Selecciona la opción TreeView, la cual mostrará una vista en forma de árbol con los distintos tipos de tornillos.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView

Tabla A.5. Diseño de Casos de Prueba RF №6

Descripción general			
Permite seleccionar un tornillo dentro del conjunto de los tornillos.			
SC 1 Seleccionar un tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción TreeView.	Selecciona la opción TreeView, la cual mostrará una vista en forma de árbol con los distintos tipos de tornillos.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView

Tabla A.6. Diseño de Casos de Prueba RF №7

Descripción general			
Permite interactuar con las herramientas de navegación sin acceder al menú.			
SC 1 Seleccionar herramientas.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción Standard Parts.	Selecciona la opción Standard Parts, la cual mostrará una ventana que tendrá los distintos campos para la conexión con la BD.	Brinda la posibilidad de establecer la conexión con la BD.	Accelerators /Standard Parts /ConfigurationDataBase

Tabla A.7. Diseño de Casos de Prueba RF №8

Descripción general			
Permite aceptar o cancelar el diálogo.			
SC 1 Aceptar o cancelar la selección.			
Escenario	Descripción	Respuesta del sistema	Flujo central



EC 1.1 Opción TreeView.	Selecciona la opción TreeView, la cual mostrará una vista en forma de árbol con los distintos tipos de tornillos.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView
EC 1.2 Opción de Aceptar.	Selecciona la opción de Aceptar.	Muestra el diseño del tornillo seleccionado en el visor de la aplicación y cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Aceptar
EC 1.3 Opción de Cancelar.	Selecciona la opción de Cancelar.	Cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Cancelar

Tabla A.8. Diseño de Casos de Prueba RF №11

Descripción general			
Permite mostrar el tornillo seleccionado en el visor de la aplicación.			
SC 1 Seleccionar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Opción View.	Selecciona la opción View la cual mostrará una serie de opciones para visualizar los tornillos en forma de tabla, de árbol y de lista.	Brinda la posibilidad de seleccionar las distintas formas de vista de los tornillos.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View
EC 1.2 Opción de TreeView.	Selecciona la opción de TreeView.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView
SC 2 Modelar el tornillo.			
Escenario	Descripción	Respuesta del sistema	Flujo central

EC 2.1 Opción TreeView.	Selecciona la opción TreeView.	Brinda la posibilidad de ver los tornillos en forma de árbol y seleccionar el tornillo que desee según la norma.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /View /TreeView
EC 2.2 Opción de Aceptar	Selecciona la opción de Aceptar	Muestra el diseño del tornillo seleccionado en el visor de la aplicación y cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Aceptar
EC 2.3 Opción de Cancelar	Selecciona la opción de Cancelar	Cierra la ventana.	Accelerators /StandardParts /ConfigurationDataBase /StdPartLibrary /Cancelar

Tabla A.9. Diseño de Casos de Prueba RF №13

Descripción general							
El sistema debe permitir la comunicación (conexión) entre las interfaces de la aplicación.							
SC 1 Realizar la comunicación entre las interfaces.							
Escenario	Descripción	Host	DBName	Username	Pass	Respuesta del sistema	Flujo central
EC 1.1 Establecer la conexión.	Se selecciona el botón Test Connection en la aplicación, si se puede acceder a la BD la conexión es satisfactoria.	localhost	BaseDatos	Admin	admin	Muestra un mensaje estableciendo la conexión de la base de datos con la aplicación.	Accelerators /StandardParts /Configuration DataBase /Test Connection/.
EC 1.2 No se establece la conexión.	Se selecciona el botón Test Connection en la aplicación, no se puede acceder a la BD.	localhost	BaseDatos	(vacío)	admin	Muestra un mensaje de que no hay conexión con la base de datos porque existen campos vacíos .	Accelerators /StandardParts /Configuration DataBase /Test Connection/.

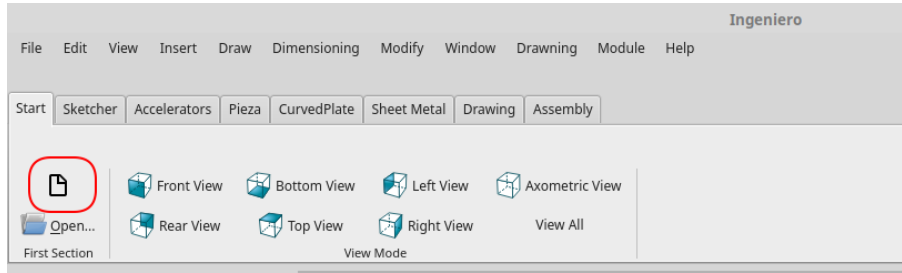


Figura A.1. Crear el nuevo visor de la aplicación



Figura A.2. Sección de los aceleradores

EC 1.3 Establecer la conexión con campos vacíos.	Se selecciona el botón Test Connection en la aplicación, dejando campos vacíos	localhost	(vacío)	Invitado	(vacío)	Detecta que existen campos vacíos y muestra un mensaje de que no hay conexión con la base de datos porque existen campos vacíos .	Accelerators /StandardParts /Configuration DataBase /Test Connection/.
--	--	-----------	---------	----------	---------	---	--

## A.2. Manual de usuario

### A.2.1. Configuración inicial

Una vez que el usuario tenga instalado el sistema DAC “Ingeniero”, y el servidor de base de datos “PostgreSQL 9.4” con el archivo *BD1.backup* (base de datos) cargado. Podrá acceder a la aplicación, donde primeramente tendrá que crear un nuevo visor de la aplicación para poder visualizar los tornillos que desee. Ver Figura A.1.

Una vez que se halla creado el nuevo visor, podrá acceder dentro de la sección de aceleradores de diseño *Accelerators*, a la sección *Standard*. Ver Figura A.2.

Al activarse el botón *Standard Parts* se muestra una ventana con diferentes opciones que permite establecer la configuración para la conexión de la base de datos ya creada con el nombre “BaseDatos”. Ver Figura A.3

- Opción Host: el usuario puede insertar un host local.

Figura A.3. Configuración para establecer la conexión

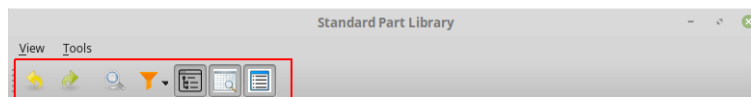


Figura A.4. Panel de herramientas

- Opción *DataBase name*: nombre de la base de datos.
- Opción *username*: nombre del usuario (Admin o Invitado).
- Opción *password*: contraseña asociada al rol establecido.
- Opción *Search DataBase*: contiene las base de datos disponibles, son reflejadas en la opción *DataBase name*.
- Opción *Test Connection*: permite establecer o no la conexión con la base de datos.

### A.2.2. Subsistemas o Módulos

Una vez que el usuario accede al sistema se muestra la vista principal del componente con el nombre *Standard Part Library*, esta interfaz cuenta con un conjunto de funcionalidades en el borde superior izquierdo. Ver Figura A.4.

- *Undo y Redo*: Permite que el usuario pueda ir hacia delante y hacia atrás dentro de los tipos de tornillos que se encuentra en el árbol ubicado en el borde izquierdo de la interfaz. Ver Figura A.5.
- *Search*: Permite al usuario buscar por el nombre la norma del tornillo que desee. Para poder realizar esta operación el usuario debe haber seleccionado dentro de los tipos de tornillos que existen en el árbol el que quiera modelar según la norma que va a buscar. Ver Figura A.6.

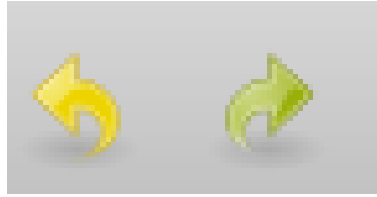


Figura A.5. Undo y Redo

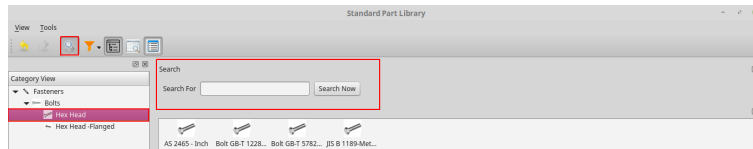


Figura A.6. Search

- *Filters*: Permite al usuario filtrar por las diferentes normas que existen en la base de datos. Los filtros que se mostrarán son solo de las normas que están definidas en la aplicación. Ver Figura A.7.
- *TreeView*: Permite al usuario ver los distintos tipos de tornillo en forma de árbol. Situado en la parte izquierda de la aplicación. Ver Figura A.8.
- *TableView*: Permite ver los datos de las normas según el tipo de tornillo y la norma seleccionada. Situada en la parte inferior derecha de la aplicación. Ver Figura A.9
- *List*: Permite al usuario ver la lista de normas existentes del tornillo seleccionado. Situada en la parte superior derecha de la aplicación. Ver Figura A.10.

Para modelar el tornillo que el usuario desee según la norma seleccionada sólo tendrá que presionar en una de las filas de la tabla correspondiente con la norma y hacer click en el botón Aceptar. Luego el tornillo se mostrará en el visor de la aplicación. Ver Figura A.11.

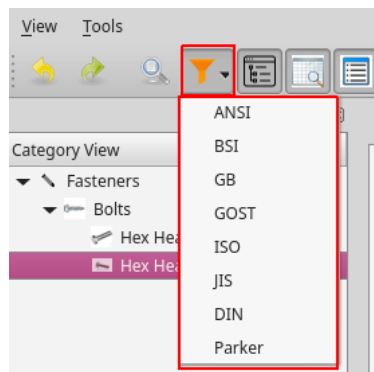


Figura A.7. Filters

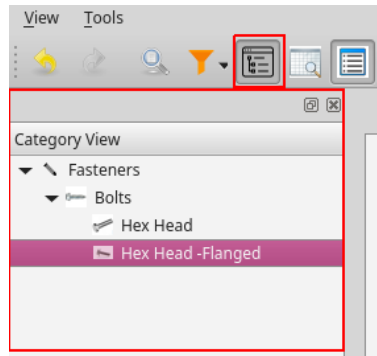


Figura A.8. TreeView

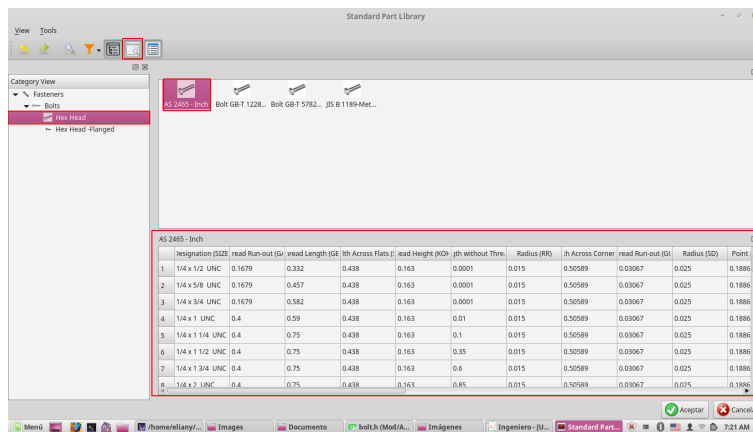


Figura A.9. TableView



Figura A.10. List

