Universidad de las Ciencias Informáticas

Facultad 4

Componente para el modelado de vigas para el Sistema CAD 2D

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

Autor:

Julio César Pérez González

Tutores:

Dr.C Augusto César Rodríguez Medina

Ing. Ángel Luis Ortega Amador

La Habana, junio de 2016

"Año 58 de la Revolución"

Declaración de autoría

Declaro ser único autor del presente trabajo de diploma y autorizo a la Universidad de las Ciencias Informáticas a ser uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2016.

_____

Julio César Pérez González

Firma del Autor

_____ _____

Dr.C Augusto Cesar Rodríguez Medina

Firma del Tutor

Ing. Ángel Luis Ortega Amador

Firma del Tutor

Resumen

El surgimiento de los sistemas de diseño asistido por computadora facilitó en la industria el trabajo de los ingenieros, diseñadores y arquitectos, dotándolos de potentes herramientas capaces de realizar labores complejas en poco tiempo y con un alto nivel de precisión y realismo. Debido a la gran importancia que tienen las vigas en la arquitectura, el presente artículo tiene como objetivo el desarrollo de un componente capaz del modelado de dichos elementos a partir de varios perfiles y del cálculo teniendo en cuenta fuerzas aplicadas sobre ellas. El componente está desarrollado con el lenguaje C++, el *framework* Qt y la tecnología *Open Cascade* para el modelado, y siguiendo la metodología de desarrollo AUP en su versión UCI. Como resultado de este trabajo se obtiene una herramienta novedosa que constituye en el país, una alternativa en software libre al empleo de sistemas propietarios e impide futuras demandas judiciales por violación de propiedad intelectual en labores de ingeniería y diseño, logrando elevar la soberanía tecnológica en ramas como la mecánica, civil y otras afines.

Palabras clave: componente, diseño, modelado, perfiles, software libre, vigas.

Introducción

La investigación y desarrollo, presentes en este documento, poseen sus bases en un proyecto de la Facultad 4 de la Universidad de las Ciencias Informáticas, el cual tiene como propósito desarrollar una aplicación para el Diseño Asistido por Computadoras (CAD, por sus siglas en inglés), destinado a los sectores de la industria nacional vinculados a la actividad de diseño; en este contexto una de las tareas fue automatizar el proceso de modelado geométrico y cálculos de las propiedades físicas de vigas utilizando las tecnologías libres *Open Cascade* y el *framework*[1] Qt.

Este documento está estructurado en tres capítulos:

- Capítulo 1: en este capítulo se aborda la fundamentación teórica de la investigación, donde se incluye un estudio de los sistemas CAD homólogos, propiedades y características de los diferentes perfiles estructurales para conformar vigas, la metodología y herramientas para el desarrollo.
- Capítulo 2: en este capítulo se abarca la propuesta de solución, se realiza el levantamiento de requisitos, el modelo del dominio, la descripción del componente, la arquitectura, los patrones de diseño, el modelo de datos, el diagrama de secuencia, el modelo conceptual y las historias de usuarios.
- Capítulo 3: en este capítulo se presentan elementos correspondientes a la implementación, como: los estándares de codificación y las pruebas realizadas al componente.

[1] (plataforma, entorno, marco de trabajo). Desde el punto de vista del desarrollo de software, un framework es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.(1)

1 Fundamentación teórica

En este capítulo se expone la situación problemática que dio origen al proceso de investigación. También se abordan temas generales sobre sistemas homólogos y aspectos técnicos sobre los diferentes perfiles estructurales utilizados para conformar vigas, así como las normas y estándares a los que pertenecen dichos perfiles. Además, se describen las diferentes tecnologías de desarrollo a utilizar.

1.1 Situación problemática

El diseño asistido por computadora (CAD, por sus siglas en inglés) ha constituido un hito para el sector de la ingeniería, la arquitectura y la construcción, especialmente porque eliminó la necesidad de dibujar los planos a mano, permitiendo además incorporar cambios con facilidad.(2)

El CAD 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. (2)

El primer sistema de gráficos fue el SAGE (*Semi Automatic Ground Environment*, en español Entorno Terrestre Semi Automático), un sistema de defensa aérea, que fue empleado para visualizar datos de radar, el cual data de los años 50, se desarrolló para las Fuerzas Aéreas de los Estados Unidos y fue desarrollado en colaboración con el MIT (*Massachusetts Institute of Technology*, en español Instituto Tecnológico de Massachusetts). (2)

En los 60, los sistemas CAD se utilizaron para diseñar espacios interiores de oficinas. En 1968 estaban ya disponibles los sistemas CAD 2D que funcionaban en sistemas de grandes ordenadores.(2)

A principios de la década del 70 varias compañías empezaron a ofrecer sistemas de diseño/dibujo automatizado. Muchos de los productos y firmas más conocidas en la actualidad tuvieron sus inicios en este periodo, por ejemplo, CATIA y CADLink. Podían encontrarse ya algunas capacidades 3D en programas de cálculo de sistemas HVAC (*Heating, Ventilation and Air Conditioning*, en español Calefacción, Ventilación y Aire Acondicionado). A finales de los años 70, un sistema típico de CAD consistía en un mini-ordenador de 16 bits con un máximo de 512 Kb de memoria y de 20 a 300 Mb de disco duro. (2)

En los años 80 AutoCAD llegó a ser el programa más popular de CAD. Muchos otros programas de compañías diversas siguieron la misma senda. Durante esta década, los programas de CAD se utilizaban básicamente para desarrollos de ingeniería. (2)

En los años 90 se generalizan las visualizaciones en 3D. A mediados de esta década aparecen muchos programas de este tipo para una gran variedad de usos y aplicaciones. A finales del siglo XX, ya los sistemas CAD se utilizaban de forma habitual, pero había una gran lucha por atraer la atención de los usuarios. Debido a esto se desarrollan mejores programas para satisfacer las necesidades crecientes de la industria y aparecen también programas sencillos de CAD. Todas estas herramientas ofrecen soluciones específicas a segmentos verticales de la industria (ingeniería civil, mecánica, etc.).(2)

En la actualidad existen compañías con varios años de experiencia que poseen productos con un gran nivel de acabado en el diseño y en el modelado, por ejemplo:

- AutoCAD e Inventor de AutoDesk Incorporated.
- Solid Edge, SolidWorks y CATIA de Dassault Systèmes.

Algunos de estos productos presentan módulos que realizan el diseño de estructuras complejas en poco tiempo y con un alto nivel de precisión y realismo. Un ejemplo de esto es el componente para el modelado de vigas a partir de un perfil estructural.

Los perfiles estructurales tienen una amplia variedad de formas y usos, de donde se obtienen las vigas que constituyen una parte importante en muchos tipos de proyectos de construcción, ya sean residenciales, comerciales o públicos. Las vigas proporcionan flexibilidad, ligereza y resistencia, además disminuyen significativamente el tiempo de construcción de las estructuras.

En Cuba el uso de software propietario que demande la autorización de sus autores (licencia) para su explotación, no es una opción viable por varias razones(3):

- El software propietario exige un pago consistente en altas sumas para la adquisición de su licencia.
- El software propietario no facilita el código fuente, por lo que no es posible corregir bugs o errores de programación.
- Para Cuba es riesgoso tener a Windows como su principal plataforma de software debido a que Microsoft está sujeta a las leyes de los Estados Unidos.
- El uso de Windows es un problema de seguridad nacional, debido a la existencia de las llamadas “puertas traseras”, que hacen posible la vigilancia remota de los sistemas por parte de órganos de inteligencia foráneos.

Por las razones anteriormente planteadas, el uso de herramientas CAD propietarias no es una opción rentable, ni segura para nuestro país por las limitaciones que esto supondría a nuestra industria.

Estas limitaciones se pudieran resolver si se utilizaran herramientas CAD software libre, como por ejemplo el LibreCAD y el FreeCAD, en sustitución al software propietario. Sin embargo, estas herramientas no cuentan con módulos que aceleren el diseño, lo que dificulta la confección de estructuras complejas como es el caso de los perfiles estructurales para conformar vigas.

Como una forma de afrontar el problema existente en el país, en la Facultad 4 de la Universidad de las Ciencias Informáticas, existe un grupo de investigación llamado “Soluciones Informáticas para la Ingeniería y la Industria” (SIPII), el cual se encuentra desarrollando una herramienta CAD que satisfaga las necesidades de los ingenieros, arquitectos, diseñadores y otros vinculados a la actividad del diseño.

Esta investigación parte de la necesidad de dicho proyecto de incluir módulos para acelerar el diseño entre los que se encuentra el componente para modelar vigas, con el objetivo de disminuir el tiempo de trabajo, aumentar la precisión y calidad de los diseños de este tipo de estructuras complejas y aportar a la industria cubana una herramienta soberana.

Teniendo en cuenta lo anteriormente expuesto se formula el siguiente **problema de investigación**:

Inexistencia del componente para automatizar el modelado de vigas en el marco de la ingeniería y el diseño a nivel nacional, como parte de una aplicación legalmente adquirida.

El **objeto de estudio** de la investigación se centra en el gráfico por computadoras para aplicaciones industriales, teniendo como **campo de acción**, el desarrollo de componentes CAD para perfiles estructurales.

Para solucionar el problema planteado se propone como **objetivo general**, desarrollar un componente para el diseño y modelado de vigas, empleando herramientas de código abierto.

A partir del objetivo general se derivan los siguientes **objetivos específicos:**

- Diseñar el componente.
- Obtener un componente funcional capaz de automatizar el proceso de diseño de perfiles estructurales para conformar vigas.

Para dar cumplimiento a los objetivos específicos se proponen las siguientes **tareas de investigación**:

- Definición del perfil y diseño de la investigación.
- Asimilación de los principales conceptos y tecnologías que se requieren para el desarrollo del componente (lenguaje C++, *framework* Qt y la tecnología *Open Cascade*)
- Estudio de las normas para perfiles estandarizados y no estandarizados.
- Obtención de requisitos a partir de soluciones similares.
- Estudio de las investigaciones previas en el marco del proyecto.
- Selección de la metodología de desarrollo.
- Selección del Sistema Gestor de Base de Datos.
- Diseño de la interfaz gráfica del componente.
- Definición de la arquitectura del componente.
- Elaboración de la estructura de clases que conforman el componente.
- Definición de los patrones de diseño.
- Modelado del flujo de datos del componente.
- Implementación del componente destinado a modelar y visualizar vigas.
- Visualización de los resultados del cálculo geométrico con las funciones de *Open Cascade*.
- Realización de pruebas al componente implementado.

Como **resultado** de la investigación se espera obtener una herramienta de software libre que permita el modelado de vigas para ser integrado al Sistema CAD 2D desarrollado por el grupo de investigación SIPII.

Para resolver y dar cumplimiento a los objetivos y las tareas propuestas se emplean un conjunto de métodos científicos los cuales se definen a continuación:

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.

Modelación: 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 CAD (FreeCAD, SolidEdge, AutoCAD Inventor), y así establecer los requisitos con las principales funcionalidades de estos.

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

1.2 Generalidades sobre perfiles estructurales para conformar vigas

En términos de diseño industrial, un perfil no es más que una forma que se define para, a partir de ella, crear elementos que sigan su mismo patrón. Los perfiles no son elementos palpables en el proceso de fabricación, pero sí definen la estructura que tendrán los objetos que se regirán por ellos. Por lo general son bocetos 2D que tienen una serie de parámetros que modifican su forma y que varían en dependencia del uso que recibirán. Los perfiles para conformar barras y tuberías (ver Figura 1), son algunos ejemplos sencillos. También existen otros tipos de perfiles que permiten conformar estructuras mucho más fuertes y estas soportan grandes cantidades de peso como es el caso de los perfiles I, C, L (ver Figura 2).(4)

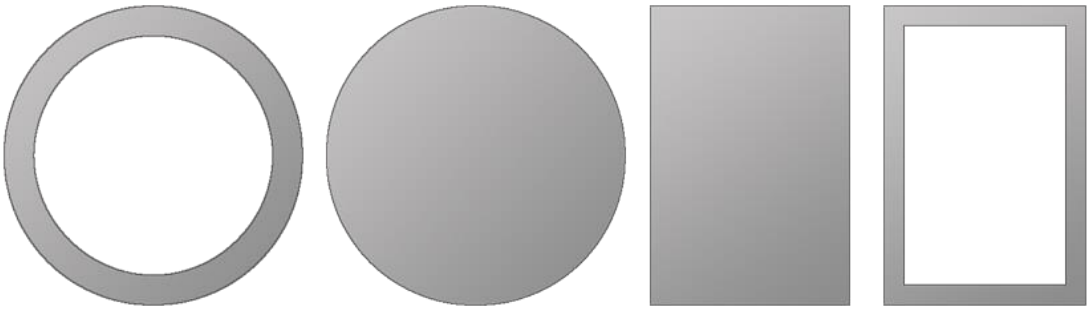


Figura 1 Ejemplo de perfiles para conformar barras y tuberías

Figura 2 Ejemplo de perfiles I,C,L.

La creación de los perfiles para barras y tuberías se realiza de manera sencilla pues se necesitan pocos parámetros como el radio de la circunferencia o el ancho y el largo, dependiendo del tipo de perfil. Por otra parte, como podemos observar en la Figura 3 Parámetros para confeccionar los perfiles, para el diseño de los perfiles I, L, C se han de tener en cuenta muchos más parámetros, por lo que podemos afirmar que crearlos manualmente consume mucho tiempo para los especialistas y los obliga a tener a mano las diferentes normas internacionales que rigen las medidas del perfil que quieran diseñar.

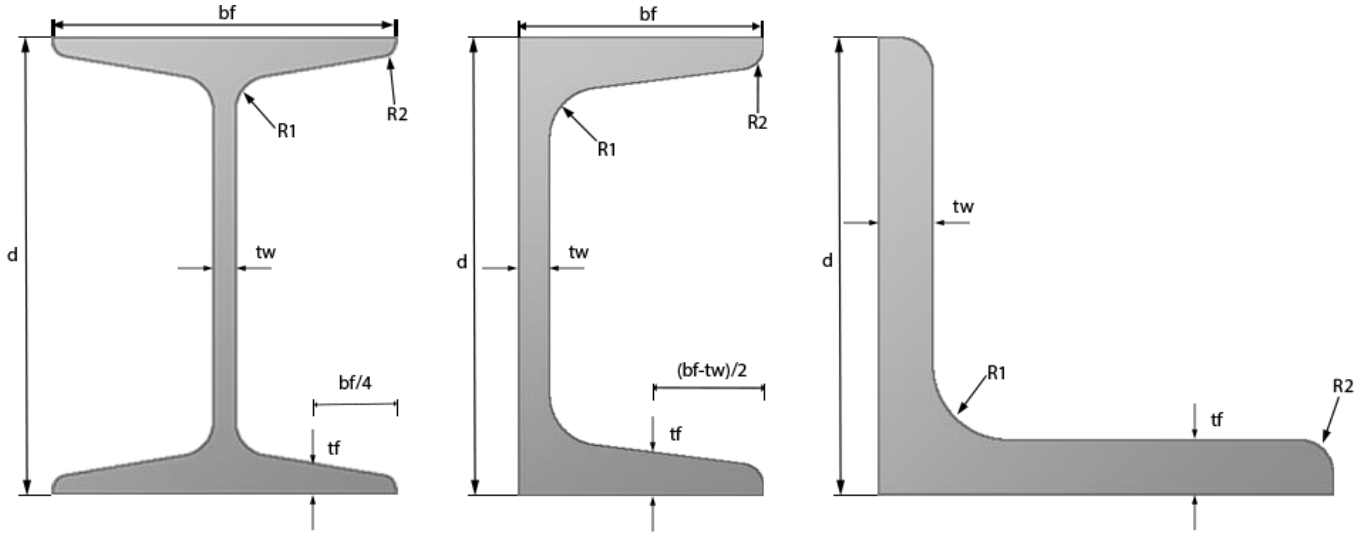


Figura 3 Parámetros para confeccionar los perfiles I,C,L

Tabla 1 Símbolos y descripciones

Símbolo	**Descripción**
d	Profundidad del perfil
bf	Ancho del perfil
tw	Espesor del alma
tf	Espesor del ala
R1	Radio del fillet[2] entre el alma y el ala
R2	Radio del fillet en el ala

[2]Fillet: Unión redondeada entre dos partes conectadas en un ángulo. (Tomado de http://www.engineering-dictionary.org)

La simbología puede cambiar dependiendo de la norma, también pueden agregarse algunos parámetros, pero básicamente conociendo los anteriores se pueden modelar los perfiles de tipo I, L y C.

Como es de esperar, estos elementos se pueden modelar en herramientas CAD donde el proceso de creación comienza con el diseño del perfil estructural, luego se aplica el proceso de extrusión y finalmente obtenemos la viga. La extrusión permite crear cubos a partir de rectángulos y cilindros a partir de círculos. Su utilización es bastante intuitiva, pues es básicamente estirar paralelamente a un eje la figura en 2D y así obtenemos un cuerpo tridimensional; sin embargo, los principios detrás de este método son más complicados, pues el algoritmo debe delimitar las aristas que pertenecen al contorno de la figura y luego las convierte en caras de manera perpendicular al plano de trabajo y así queda conformada la figura 3D.(4)

1.2.1 Normas internacionales de perfiles estructurales

Según la *International Standard Organization* (ISO, en español Organización Internacional de Normalización), “un estándar o norma es un documento que provee requerimientos, especificaciones, directrices o características que pueden ser utilizadas consistentemente para garantizar que los materiales, productos, procesos o servicios están aptos para su propósito”.(5)

La estandarización o normalización es la actividad que tiene por objeto establecer, ante problemas reales o potenciales, disposiciones destinadas a usos comunes y repetidos, con el fin de obtener un nivel de ordenamiento óptimo en un contexto dado, que puede ser tecnológico, político o económico.(5)

La utilización de normas en las diferentes esferas brinda varios beneficios para las empresas y la sociedad, por ejemplo (6):

- Aseguran la competencia leal y brindan estímulos para mejorar la calidad.
- Ayudan a optimizar las operaciones, a disminuir costos, a incrementar la satisfacción de los clientes y mejoran en general la productividad y la competitividad.
- Las normas evitan barreras comerciales innecesarias y facilitan el acceso a los mercados mundiales.
- Los consumidores pueden tener la confianza que los productos y servicios son seguros, confiables y de calidad adecuada.

Los estándares abarcan la mayoría de los procesos, productos y servicios que se realizan a nivel mundial. El diseño mecánico es una rama de la industria que no está exenta de este proceso de normalización. Algunas de las entidades normalizadoras más conocidas internacionalmente que están relacionadas con esta rama son ANSI (Instituto Nacional Estadounidense de Estándares), DIN (Instituto Alemán de Normalización) y la ISO (Organización Internacional para la Estandarización).

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 materiales utilizados en la industria, incluyendo perfiles estructurales para conformar vigas. La mayoría de los sistemas CAD existentes en la actualidad se rigen también por estas normas para el diseño mecánico.

En la Figura 4 se muestra una tabla de dimensiones de acuerdo con la norma ASTM A6M utilizada por la empresa Nucor-Yamato Steel para la creación de vigas con perfil C.(7)

Prime Section Group	Section Size	Area	Depth	Web Thick-ness	Flange Width	Flange Thick-ness	Distance			Fillet Radius	Elastic Properties						Plastic Modulus	
											X - X			Y - Y				
	mm x kg/m	A mm^2	d mm	t_w mm	b_f mm	t_f mm	T mm	k mm	k_1 mm	R mm	I_x $10^6\ mm^4$	S_x $10^3\ mm^3$	r_x mm	I_y $10^6\ mm^4$	S_y $10^3\ mm^3$	r_y mm	Z_x $10^3\ mm^3$	Z_y $10^3\ mm^3$
380 x 85	C380 x 74	9 480	381	18.2	94	16.5	319	31	22	13	168	882	133.1	4.6	62	22.0	1 123	133
	C380 x 60	7 610	381	13.2	89	16.5	319	31	20	13	144.8	762	138.4	3.8	55	22.4	942	112
	C380 x 50.4	6 430	381	10.2	86	16.5	319	31	18	13	131.1	688	142.7	3.4	51	22.9	832	101
310 x 75	C310 x 45	5 690	305	13.0	80	12.7	257	24	17	10	67.4	442	109.0	2.1	34	19.4	554	71
	C310 x 37	4 740	305	9.8	77	12.7	257	24	15	10	59.9	393	112.5	1.9	31	19.8	482	63
	C310 x 30.8	3 930	305	7.2	74	12.7	257	24	14	10	53.7	352	117.1	1.6	28	20.2	420	57

Figura 4 Parámetros de creación de vigas tipo C.(7)

Los estándares no solo son utilizados por las empresas que se dedican a la fabricación, por tal motivo algunas herramientas CAD incluyen en sus módulos el tratamiento de los estándares para facilitar el trabajo de los diseñadores.

1.3 Herramientas y componentes para el modelado y cálculo de vigas

En la actualidad existen componentes asociados a sistemas CAD que sirven para el modelado de vigas y cálculo de sus propiedades físicas, con el fin de brindar resultados precisos a dichos cálculos y acelerar el proceso de diseño llevado a cabo por ingenieros, diseñadores y arquitectos. También existen herramientas independientes a los sistemas CAD que realizan diferentes cálculos sobre las vigas. En este apartado se estudiarán algunas de estas herramientas desarrolladas para diferentes plataformas y con diferentes tecnologías.

1.3.1 Módulos existentes en sistemas para el diseño asistido por computadora

Autodesk Inventor

Autodesk Inventor proporciona soluciones de ingeniería y diseño de grado profesional para diseño mecánico en 3D, simulación, herramientas, 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.(8)

El módulo para el diseño de vigas a partir de perfiles en Autodesk Inventor en su versión del 2009 llamado *Beam and Column Calculator* permite seleccionar el tipo de perfil que queremos modelar, introducir los datos, seleccionar el tipo de material, tanto estándar, como personalizado por el usuario. Permite también insertar cargas y soportes de diferentes tipos. Los cálculos se realizan en forma de viga o de columna. El resultado de algunos cálculos se muestran en gráficos que ayudan a una mejor comprensión.(9)

FreeCAD

FreeCAD es un modelador CAD 3D completamente de código abierto (Licencia LGPL[3]). FreeCAD está dirigido directamente a la ingeniería mecánica y diseño de productos, pero también se emplea en una amplia gama de usos en la ingeniería, como la arquitectura o de otras especialidades de la ingeniería.(10)

Solid Edge

[3] GNU Lesser General Public License. Licencia Pública General de GNU. Permite que se utilice gratuitamente con fines comerciales.

Solid Edge es un portafolio de herramientas de software asequibles y fáciles de usar que abordan todos los aspectos del proceso de desarrollo de productos - diseño en 3D, simulación, fabricación, gestión del diseño y mucho más gracias a un creciente ecosistema de aplicaciones. Solid Edge combina la velocidad y simplicidad del modelado directo con la flexibilidad y el control de diseño paramétrico hecho posible con synchronous technology.(11)

CATIA (*computer-aided three dimensional interactive application*)

CATIA (computer-aided three dimensional interactive application, en español Aplicación Interactiva Tridimensional Asistida por Ordenador) es un programa informático de diseño, fabricación e ingeniería asistida por computadora comercial realizado por *Dassault Systèmes*. El programa está desarrollado para proporcionar apoyo desde la concepción del diseño hasta la producción y el análisis de productos. Está disponible para *Microsoft Windows*, Solaris, IRIX y HP-UX. Provee una arquitectura abierta para el desarrollo de aplicaciones o para personalizar el programa. Las interfaces de programación de aplicaciones, se pueden programar en Visual Basic y C++. Fue inicialmente desarrollado para servir en la industria aeronáutica, se ha hecho un gran hincapié en el manejo de superficies complejas.(12)

1.3.2 Herramientas independientes para el cálculo de vigas

Con el objetivo de proporcionar ayuda a los diseñadores, ingenieros mecánicos y arquitectos se han creado disímiles herramientas que cubren necesidades específicas para el cálculo de vigas. Algunas herramientas independientes estudiadas en el transcurso de la investigación fueron:

MITCalc

“Es un conjunto de cálculos técnicos, ingenieriles e industriales para las rutinas diarias. Es preciso, confiable y te guiará rápidamente a través del diseño de componentes, la solución a un problema técnico, o un cálculo de un punto de la ingeniería, sin necesidad de poseer conocimiento experto”.(13)

MITCalc contiene tanto el diseño como la verificación de los cálculos de muchas tareas comunes, tales como: engranajes rectos, engranaje helicoidal, resortes, viga, tolerancias, análisis de la tolerancia, fórmulas técnicas y muchos otros.(13)

“Es un sistema abierto diseñado en Microsoft Excel a la que usted puede hacer futuras modificaciones o extensiones de usuario sin tener habilidades de programación”.(13)

Permite el intercambio de datos entre muchos sistemas CAD 2D y 3D, por ejemplo AutoCAD, AutoCAD LT, IntelliCAD, Autodesk Inventor, SolidWorks, SolidEdge, Pro/Engineer, entre otros.(13)

eFunda

Es un sitio web que responde a una empresa con el mismo nombre y tiene como objetivo crear un destino en línea para la comunidad de ingeniería, donde pueda encontrarse ayuda en la solución de problemas complejos de diseño de forma concisa y confiable. Abarca materiales de nivel universitario de las escuelas de ingeniería, además dice exactamente bajo qué condiciones son utilizadas las fórmulas que posee.(14)

Bendingmomentdiagram

Bendingmomentdiagram pertenece al conjunto de herramientas de la empresa SkyCiv, que se encarga de proveer software para análisis estructural en línea.(15)

“Bendingmomentdiagram.com es una calculadora en línea gratuito que genera diagramas de momento flector y fuerza cortante para la mayoría de vigas simples. La calculadora es completamente personalizable para adaptarse a la mayoría de las vigas; que es una característica no disponible en la mayoría de otras calculadoras”(16). Ofrece también otras herramientas y tutoriales.

Para acceder a otras funcionalidades y herramientas de SkyCiv es necesario efectuar pagos mensuales.(15)

XVIGAS

“Programa orientado a ámbitos educativos que permite resolver problemas de vigas. No es un programa de cálculo de estructuras, sino un asistente de aprendizaje, ayuda en la resolución de ejercicios y prácticas”.(17)

Es un programa desarrollado en C, libre, de código fuente abierto, y gratuito que funciona en Windows.(17)

BEAMANAL

Hoja de cálculo que provee los cálculos necesarios para determinar la distribución de cargas internas (distribución de momentos y fuerza cortante) junto con la distribución de la pendiente y la deflexión. Es principalmente usado para estructuras de acero en el mundo de la ingeniería civil.(18)

Es una herramienta de libre uso y se le pueden realizar cambios.

1.4 Metodología para el desarrollo de software

Como guía para el desarrollo del trabajo de diploma fue utilizada la metodología Proceso Unificado Ágil (Agile Unified Process, AUP).

AUP es un enfoque de modelado híbrido creado por Scott Ambler cuando combinó el *Rational Unified Process* (RUP) con los métodos ágiles (A.M). Mediante la combinación de RUP con AM, Ambler creó un marco sólido de procesos que se puede aplicar a todo tipo de proyectos de software, grandes o pequeños.(19)

Los principios que sustentan AUP son (19):

- 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) se realizó una variación a dicha metodología con el propósito de normalizar el proceso de desarrollo de software dentro de todos sus centros productivos.(20)

De las 4 fases que define originalmente AUP, la variación AUP-UCI define:

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 Construcción Transición	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.
	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 5 Descripción de las fases AUP y AUP-UCI (20)

AUP-UCI también define cuatro escenarios para modelar el sistema en los proyectos productivos(20):

- 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.
- 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.

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.

1.5 Herramientas y tecnologías

El desarrollo continuo de las TICs propicia el perfeccionamiento de las herramientas informáticas, por lo que se hace necesario realizar una investigación acerca de las posibles herramientas a utilizar para el desarrollo del componente. Seguidamente se describen las tecnologías y herramientas empleadas en la presente investigación, realizando un análisis de las principales características de cada una de ellas.

1.5.1 Open CASCADE Technology

Open CASCADE *Technology* (OCCT) es una biblioteca de clases orientada a objetos desarrollada en C++ y diseñada para la producción de sofisticadas aplicaciones CAD/CAM[4]/CAE[5]. Provee servicios para superficies 3D y modelado de sólidos, intercambio de datos CAD y visualización.(21)

OCCT es una tecnología software libre, puede ser redistribuida y/o modificada bajo los términos de LGPL versión 2.1, con excepciones adicionales. Está diseñada para ser altamente portátil y es conocida por trabajar en una amplia gama de plataformas (UNIX, Linux, Windows, Mac OS X, Android).(21)

Open CASCADE cuenta con una comunidad que ha desarrollado Open CASCADE *Community Edition* (OCE), 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.(22)

Se decide emplear la biblioteca OCE debido a las potencialidades que ofrece para el desarrollo de herramientas de tipo CAD/CAM/CAE y su actual utilización en el Sistema CAD 2D, sistema al que está destinado la presente investigación.

1.5.2 Framework de desarrollo

Qt es un *framework* de desarrollo para aplicaciones multiplataforma que simplifica mucho el desarrollo de aplicaciones en C++ de forma nativa, también puede ser utilizado en otros lenguajes

[4] Computer Aided Manufacturing (Fabricación Asistida por Computadora).

[5] Computer Aided Engineering (Ingeniería Asistida por Computadora).

y tiene un amplio apoyo. Es una biblioteca para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de consola y servidores.(23)

“Provee módulos para el desarrollo en áreas como redes, bases de datos, OpenGL, tecnologías web, sensores, protocolos de comunicación, procesamiento de XML y JSON, impresión, generación de PDF, entre otros”.(23)

Algunas características que posee el *framework* Qt son (23):

- Constituye además una biblioteca para la creación de interfaces gráficas. Se distribuye bajo una licencia libre GPL (o QPL), además de la LGPL, que permite su utilización gratuita con fines comerciales.
- Se encuentra disponible para un gran número de plataformas: Linux, MacOs X, Solaris, HP-UX, UNIX con X11, Windows.
- Compatibilidad multiplataforma con un sólo código fuente.
- Fácil de internacionalizar.
- Arquitectura lista para plugins.

Qt se escoge como *framework* de desarrollo debido a todas las ventajas antes mencionadas, además es el utilizado en el FreeCAD que es la herramienta CAD de mayor desarrollo dentro del software libre y que constituye la guía para el desarrollo del Sistema CAD 2D.

1.5.3 Lenguaje de programación

C++ es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C, abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos por lo que se considera un lenguaje híbrido multiparadigma. (16)

C++ es un lenguaje de programación maduro y de gran velocidad de compilación y presenta las siguientes características(24):

- Lenguaje versátil, potente y general.
- Lenguaje rico en operadores y expresiones.
- Flexible, conciso y eficiente.
- Presenta programación modular.
- Introduce nuevas palabras claves y operadores para manejo de clases.

C++ está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel. Además, se trata de un lenguaje de programación estandarizado (ISO/IEC 14882:1998), ampliamente difundido, y con una biblioteca estándar que lo ha convertido en un lenguaje universal, de propósito general, y ampliamente utilizado tanto en el ámbito profesional como en el educativo. (16)

Posee una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel(24):

- Posibilidad de redefinir los operadores (sobrecarga de operadores).
- Identificación de tipos en tiempo de ejecución (RTTI).

Se escoge C++ como lenguaje de programación debido a las potencialidades expuestas anteriormente, además de que la biblioteca de visualización OCE está desarrollada sobre este lenguaje y el *framework* Qt define C++ como su lenguaje nativo.

1.5.4 Entorno de Desarrollo Integrado

El *framework* Qt incluye su propio Entorno de Desarrollo Integrado (IDE) llamado Qt Creator. También es multiplataforma y ofrece completamiento inteligente de código, resaltado de sintaxis y un sistema de ayuda integrado, depurador y la integración de perfiles y también la integración de sistemas de control de versiones (por ejemplo, git, Bazaar).(23)

1.5.5 Herramienta de modelado

Visual Paradigm para UML 8.0 es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de *Software*, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos.(25)

Visual Paradigm también ofrece(25):

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Documentación automática Ad-hoc.
- Ambiente visualmente superior de modelado.
- Sofisticado diagramador automáticamente de *layout*.
- Sincronización de código fuente en tiempo real.

1.5.6 Lenguaje de modelado

“UML son las siglas de *Unified Modeling Language* (Lenguaje Unificado de Modelado), notación con que se construyen sistemas por medio de conceptos orientados a objetos”.(26)

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.(26)

1.5.7 Sistemas Gestores de Bases de Datos

Algunas de las herramientas CAD mencionadas en secciones previas poseen una base de datos donde almacenan las dimensiones estándares de diferentes elementos utilizados con frecuencia en el diseño mecánico, por ejemplo, resortes, materiales, perfiles estructurales para conformar vigas; con el objetivo de lograr una mayor productividad y acelerar el trabajo de los diseñadores e ingenieros mecánicos, además de eliminar documentación y posibles errores humanos al introducir los datos.

Teniendo en cuenta las ventajas descritas en el subepígrafe 1.2.1 que ofrece el uso de estándares se decide realizar una investigación de algunos de los Sistemas Gestores de Bases de Datos (SGBD) más conocidos para seleccionar el más adecuado para el desarrollo de la investigación.

PostgreSQL

PostgreSQL es un SGBD objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema, garantizando que un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.(27)

Se encuentra disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit. Algunas de las características más notables de PostgreSQL son: estabilidad, potencia, robustez, facilidad de administración e implementación de estándares.(27)

MySQL

MySQL es un sistema gestor de bases de datos (SGBD) de código abierto y ampliamente usado por su simplicidad y notable rendimiento.(28)

Entre sus principales características podemos encontrar:(28)

- Está desarrollado en C/C++.
- La API se encuentra disponible en C, C++, Eiffel, Java, Perl, PHP, Python, Ruby y TCL.
- Está optimizado para equipos de múltiples procesadores.
- Es muy destacable su velocidad de respuesta.
- Se puede utilizar como cliente-servidor o incrustado en aplicaciones.
- Cuenta con un rico conjunto de tipos de datos.
- Es altamente confiable en cuanto a estabilidad se refiere.

SQLite

SQLite es un SGBD cuyo código es de dominio público y por lo tanto es libre para el uso para cualquier propósito, comercial o privado. Es el SGBD de mayor despliegue en el mundo con más aplicaciones de las que podemos contar, incluyendo varios proyectos de alto perfil.(29)

Entre sus principales características podemos encontrar(29):

- Motor de base de datos SQL incrustado, o sea, no tiene un proceso de servidor independiente.
- Lee y escribe directamente en archivos de disco ordinarios.
- Es probado con mucho cuidado antes de cada lanzamiento y tiene una reputación de ser muy fiable.
- El formato de archivo de la base de datos es multiplataforma.
- Biblioteca compacta con un tamaño menos a los 500 Kb con todas las características habilitadas.

Fundamentación del SGBD seleccionado

Debido a que los sistemas CAD de por sí consumen valiosos recursos en las estaciones de trabajo (memoria RAM, procesador, almacenamiento) se escoge para el desarrollo de este trabajo el sistema gestor de bases de datos SQLite3, debido a todas las características que posee, siendo las de mayor peso el hecho de ser un motor de base de datos SQL incrustado y poseer una biblioteca compacta.

1.6 Conclusiones parciales

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

- La utilización de sistemas CAD propietarios en la industria cubana presentan limitaciones que impiden su empleo de manera institucional. En cuanto a las herramientas libres, la investigación arrojó que no poseen módulos que faciliten el proceso de diseño, específicamente el de perfiles estructurales para conformar vigas. De las herramientas independientes estudiadas para realizar cálculos sobre las vigas se determinó que no presentan todas las funcionalidades deseadas.
- Las herramientas, lenguaje, metodología y tecnologías definidas en el ambiente de desarrollo permitieron definir el perfil tecnológico de la investigación.

2 Propuesta de solución

En el presente capítulo se aborda la propuesta de solución e identifican y describen los requisitos funcionales y no funcionales. Se presenta el estilo y patrón arquitectónico, los patrones del diseño empleados, el diagrama de clases y las historias de usuario. Además, se aborda la metodología de cálculo a emplear.

2.1 Modelo conceptual

El modelo conceptual (o modelo de dominio) es una representación visual de los conceptos u objetos del mundo real significativos para un problema. Representa clases conceptuales del dominio del problema. Así como conceptos del mundo actual, no de los componentes de software.(26)

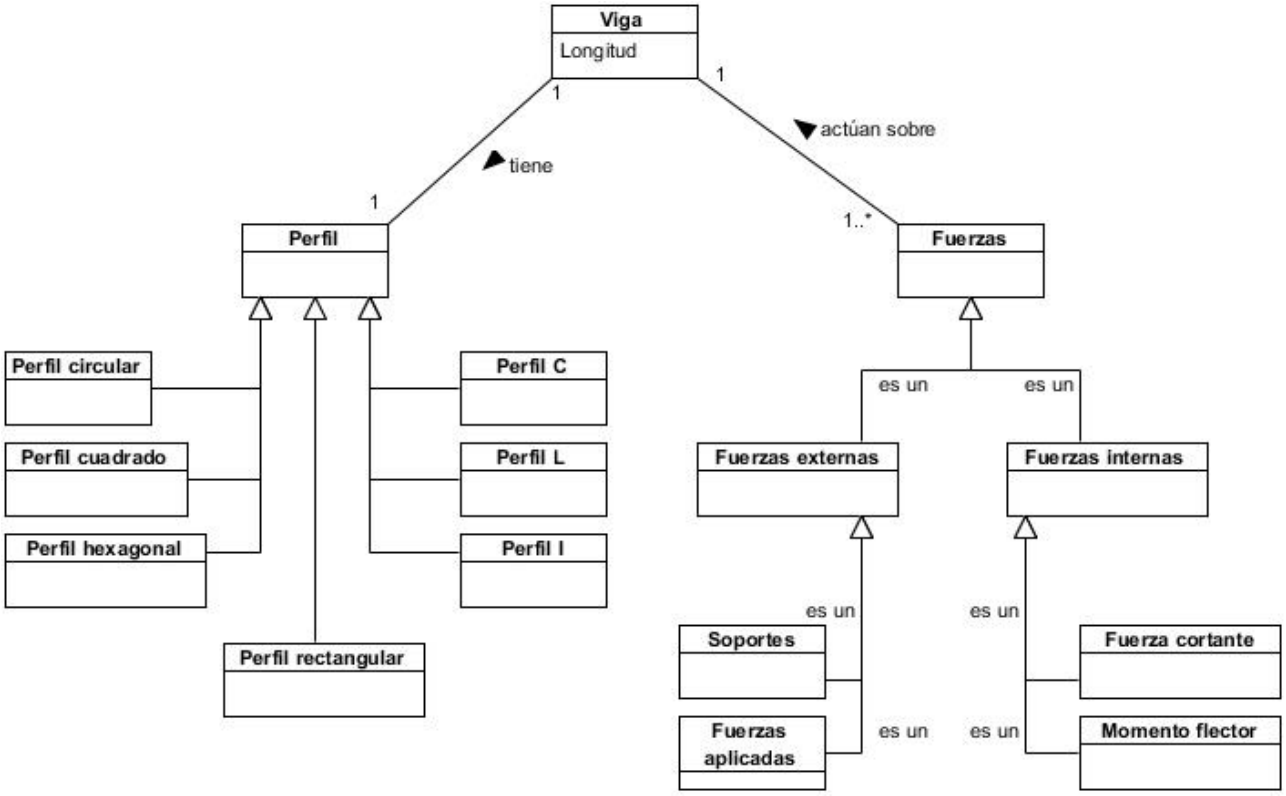


Figura 6 Modelo del dominio

2.2 Principales conceptos relacionados a la investigación

Perfiles estructurales: “los perfiles son barras de distintas formas de sección que adoptan los elementos longitudinales de una estructura, generalmente metálicos, para adaptarse lo más adecuadamente posible a su función y a los esfuerzos que les son requeridos”.(30)

Viga: “usualmente son elementos prismáticos rectos y largos diseñados para soportar cargas aplicadas en varios puntos a lo largo del elemento”.(31)

Fuerza cortante: se produce cuando una fuerza perpendicular se aplica a una viga. Estas fuerzas se producen a lo largo de numerosos puntos y es importante determinar donde estos cortes alcanzan los puntos más altos ya que puede ser donde falle la viga.(31)

Momento Flector: se produce cuando se aplica una fuerza a una distancia determinada de un punto de referencia; causando un efecto de flexión. En términos más simples, un momento flector es básicamente una fuerza que hace que algo se doble. Si no es bien retenido el objeto, la fuerza de flexión hará que el objeto gire alrededor de un punto determinado.(31)

Soporte: apoyo o sostén.(32)

Fuerzas externas: representan la acción que ejercen otros cuerpos sobre el cuerpo rígido en consideración. Son las responsables del comportamiento externo del cuerpo rígido. Las fuerzas externas causan que el cuerpo se mueva o aseguran que éste permanezca en reposo.(31)

Fuerzas internas: son aquellas que mantienen unidas las partículas que conforman al cuerpo rígido. Si éste está constituido en su estructura por varias partes, las fuerzas que mantienen unidas a dichas partes también se definen como fuerzas internas.(31)

Diseño: proceso de aplicar las diversas técnicas y principios científicos conocidos, con el propósito de definir objetos o sistemas de naturaleza mecánica, como son: piezas, estructuras, mecanismos, máquinas, dispositivos e instrumentos diversos.(33)

Modelado: proceso mediante el cual se genera una idealización matemática que pretende representar la conducta real de la estructura a ser construida.(34)

2.3 Descripción de la propuesta de solución

El componente contará con 2 interfaces principales: la interfaz de modelado y la de cálculo. La interfaz de modelado estará dividida en 2 vistas: modelado personalizado y modelado normalizado de las vigas. Ambas partes poseerán un botón *Cancel* que cierre la ventana, un botón *Create* que dibuje la viga a partir del perfil seleccionado y un botón *Calculate*, el cual muestre la interfaz para el cálculo de la viga seleccionada.

Inicialmente se muestra la interfaz de modelado normalizado con los datos de los perfiles estándares extraídos de la base de datos. La información de los estándares estará dividida en tres campos: el nombre del estándar, las diferentes familias que componen al estándar seleccionado y

las diferentes medidas de perfiles que componen a la familia seleccionada, además habrá un campo para introducir la longitud con la que se creará la viga. Esta vista poseerá un botón *Custom* que de paso a la vista de modelado personalizado.

La vista de modelado personalizado permitirá seleccionar el tipo de perfil a partir del que se desee crear una viga, también muestra una imagen de apoyo con el tipo de perfil y sus propiedades. Esta vista poseerá un botón *Standard* que de paso a la vista de modelado normalizado explicado anteriormente.

La interfaz de cálculo será capaz de añadir diferentes tipos de cargas teniendo en cuenta su fuerza y su posición con respecto al origen de la viga. También, se podrán incluir distintos tipos de apoyos en diferentes posiciones. Esta interfaz constará con una sección para mostrar los resultados de los cálculos realizados.

2.4 Requisitos

Los requisitos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requisitos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema como el control de un dispositivo, hacer un pedido o encontrar información.(35)

Los requisitos funcionales permiten conocer detalladamente las características y funciones del sistema. Los requisitos no funcionales dan a conocer las cualidades que deberá cumplir el sistema.

2.4.1 Requisitos funcionales

RF 1. Modelar la viga a partir de un perfil estructural personalizado.

- RF 1.1. Modelar la viga a partir del perfil circular.
- RF 1.2. Modelar la viga a partir del perfil rectangular.
- RF 1.3. Modelar la viga a partir del perfil cuadrado.
- RF 1.4. Modelar la viga a partir del perfil hexagonal.
- RF 1.5. Modelar la viga a partir del perfil C.
- RF 1.6. Modelar la viga a partir del perfil L.
- RF 1.7. Modelar la viga a partir del perfil I.

RF 2. Modelar la viga a partir de un perfil estructural normalizado.

RF 3. Realizar cálculos en la viga creada a partir del perfil seleccionado.

- RF 3.1. Insertar cargas puntuales.
- RF 3.2. Eliminar cargas.

RF 3.3. Insertar soportes.

RF 3.4. Eliminar soportes.

RF 3.5. Mostrar resultados de los cálculos.

2.4.2 Requisitos no funcionales

Portabilidad:

RNF 1. Instalabilidad: se debe poder instalar en un sistema operativo basado en GNU-Linux de 64bits.

Funcionalidad:

RNF 2. Precisión: debe poseer una alta precisión, en cuanto a la cantidad de cifras significativas, en los datos introducidos y mostrados.

Usabilidad:

RNF 3. Comprensibilidad: debe poseer imágenes para una mejor compresión de las variables.

2.4.3 Historias de usuario

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.(36)

Un ejemplo de historia de usuario realizada en la propuesta de solución:

Tabla 2 HU_Modelar la viga a partir del perfil rectangular.

Número: 3	**Nombre del requisito:** Modelar la viga a partir del perfil rectangular
Programador: Julio César Pérez González	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 7 días
Riesgo en Desarrollo: N/A	**Tiempo Real:** 7 días
Descripción:	

1- Objetivo:

Permitir la modelación de la viga a partir del perfil rectangular.

2- Acciones para lograr el objetivo (precondiciones y datos):

Para modelar una viga con perfil rectangular hay que:

- Tener en cuenta los datos: Altura, ancho, longitud de la sección, ahuecado y grosor.
- El tipo de sección debe tener valor “Rectangular”. (Ver HU_Modelar la viga a partir de un perfil estructural personalizado)

3- Comportamientos válidos y no válidos (flujo central y alternos):

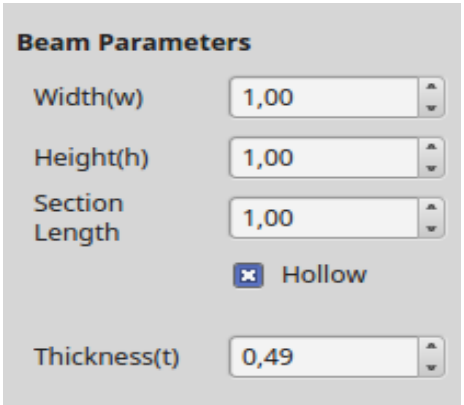
- Los campos altura, ancho, longitud de la sección y grosor deben poseer valores numéricos reales positivos.
- El grosor debe tener un valor menor que los valores de la altura y el ancho.
- Ahuecado: Campo de selección.

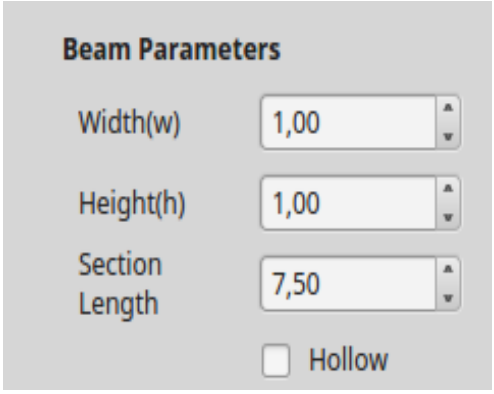
4- Flujo de la acción a realizar:

- El usuario introduce los valores correspondientes a la altura, el ancho y la longitud de la sección.
- Si se marca el campo de acción Ahuecado se muestra el campo para insertar el valor del grosor.
- Si el valor del grosor está incorrecto se muestra una alerta y señalará el campo en cuestión dando la posibilidad al usuario de realizar nuevamente la acción.
- Al presionar el botón “*Create*” se visualiza la viga de perfil rectangular. (ver HU_Modelar la viga a partir de un perfil estructural personalizado)
- Si el usuario presiona el botón “*Cancel*” se cerrará la ventana. (ver HU_Modelar la viga a partir de un perfil estructural personalizado)

Observaciones:

Prototipo de interfaz





2.5 Diseño

El diseño está definido según la IEEE[6] de dos formas: “el proceso para definir la arquitectura, los componentes, las interfaces y otras características del sistema” y “el resultado del proceso (mencionado)”. (37)

Teniendo en cuenta la segunda definición, el resultado del diseño debe describir la arquitectura utilizada, que es como el software se descompone y organiza; además de las interfaces y la descripción detallada de los componentes que posibiliten su posterior desarrollo. (37)

2.5.1 Arquitectura de software

La arquitectura del software es la estructura u organización de los componentes (módulos), la manera en que estos interactúan y la estructura de datos utilizada.(38)

Estilo arquitectónico

Un estilo arquitectónico es una transformación impuesta al diseño de todo un sistema y su objetivo es establecer una estructura para todos los componentes de dicho sistema.(38)

La familia de estilos arquitectónicos seleccionada es Llamada y retorno pues permite que un diseñador de software obtenga una estructura de programa que resulta relativamente fácil modificar y cambiar de tamaño. Miembros de esta familia son las arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objeto y los sistemas jerárquicos en capas.(38)

Sistemas jerárquicos en capas

En (39) los autores adoptan el concepto de estilo en capas como una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior.

La utilización del estilo en capas propicia ciertas ventajas, a las que se incluyen las que hereda de la familia de estilos Llamada y retorno que son modificabilidad[7] y escalabilidad[8]. Estas ventajas son (39):

---

[6] Institute of Electrical and Electronics Engineers.

[7] Es un atributo de calidad que tiene que ver con el costo del cambio y la facilidad con que un sistema se acomoda a los cambios. (También llamado mantenibilidad o portabilidad por ISO).

[8] La capacidad que tiene un sistema informático de modificar su configuración o su tamaño, para ajustarse a los cambios.(40)

- Diseño basado en niveles de abstracción creciente, lo que facilita la división de problemas complejos en una secuencia de pasos incrementales.
- Admite optimizaciones y refinamientos.

En la Figura 7 podemos observar una propuesta de arquitectura para el módulo aprovechando las ventajas que nos brinda el estilo arquitectónico en capas.

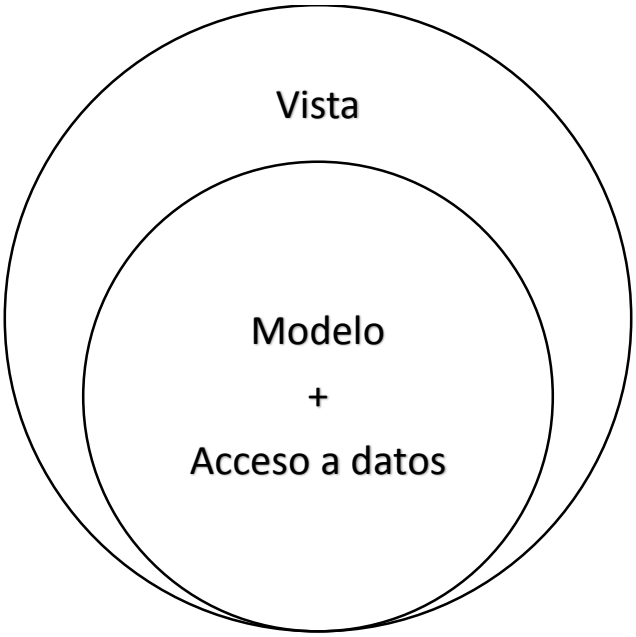


Figura 7 Propuesta de arquitectura.

La capa Vista contendrá las clases interfaces con las que el usuario interactúa. La capa Modelo + Acceso a datos contendrá las clases que permiten la creación de los objetos (vigas y fuerzas) por la parte del Modelo y la clase que permite la conexión a la base de datos por la parte de Acceso a datos. Para un mayor entendimiento de la distribución de las clases dentro de las capas ver Figura 8 Diagrama de clases en el siguiente apartado.

2.5.2 Modelo de clases del componente

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación.(26)

En la Figura 8 se muestra el diagrama de clases correspondiente al componente.

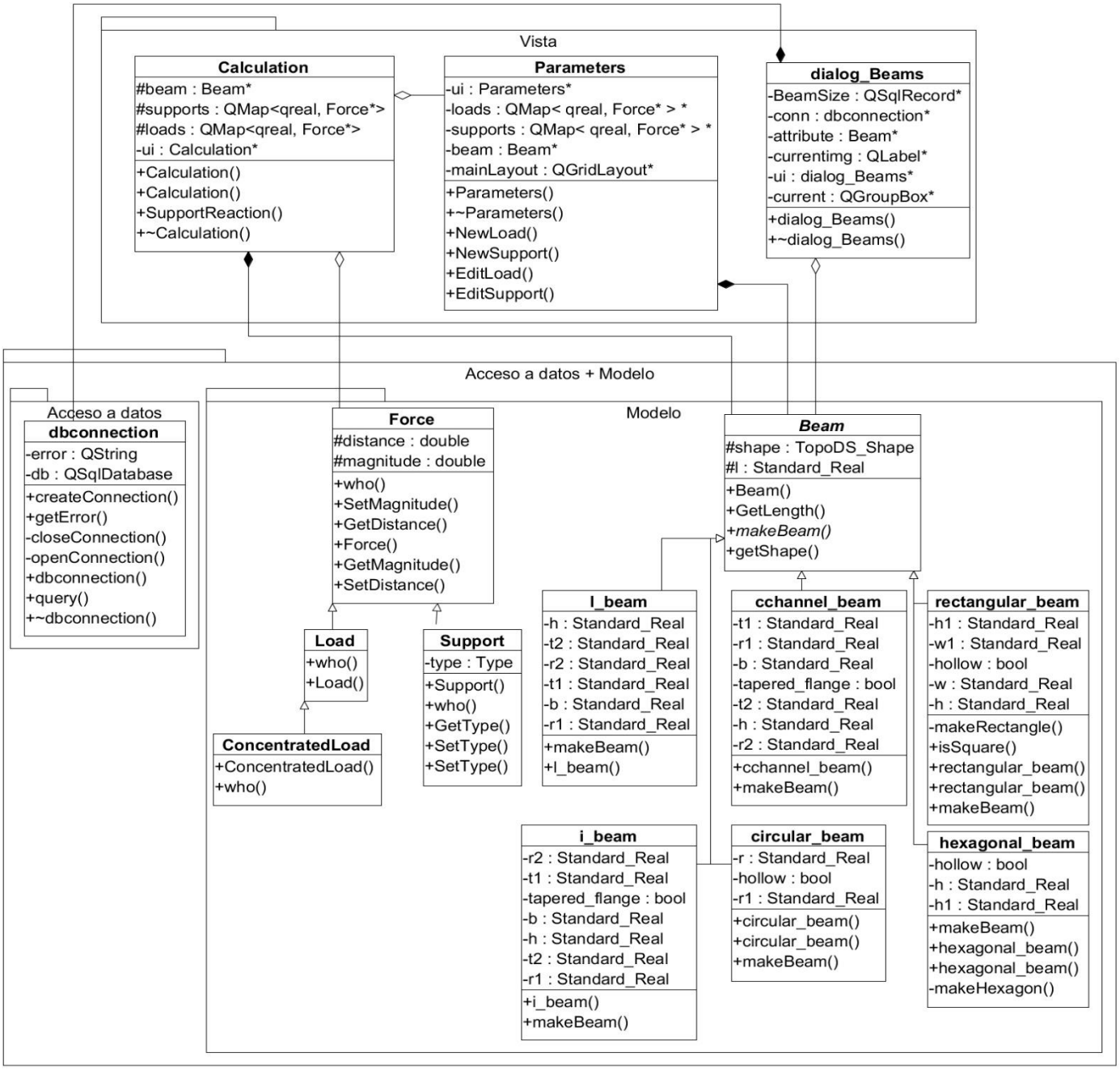


Figura 8 Diagrama de clases.

2.5.3 Patrones de diseño

Según Craig Larman un patrón es una pareja de problema/solución con un nombre y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas.(26)

Los patrones GRASP[9] describen los principios fundamentales de la asignación de responsabilidades a objetos.(26)

Los patrones GRASP definidos en el modelo de clases del componente fueron:

Experto: Asignar una responsabilidad al experto en información: la clase que posee la información necesaria para cumplir con la responsabilidad.

Creador: Permite la creación de objetos de una clase determinada. Es utilizado en las clases controladoras.

Polimorfismo: Se usa cuando varía el tipo de alternativas o comportamientos relacionados, permite asignar la responsabilidad del comportamiento a los tipos en que varía el comportamiento. Se evidencia en las clases rectangular_beam, l_beam, i_beam, hexagonal_beam, circular_beam y cchannel_beam que son tipos de beam.

2.5.4 Diagramas de secuencia del diseño

"Un diagrama de secuencia representa una interacción como un gráfico bidimensional. La dimensión vertical es el eje de tiempo, que avanza hacia abajo de la página. La dimensión horizontal muestra los roles de clasificador que representan objetos individuales en la colaboración".(41)

"Los diagramas de secuencia dan una descripción grafica de las interacciones del actor y de las operaciones a que da origen".(26)

A continuación, se muestran los diagramas de secuencia de la historia de usuario: Modelar la viga a partir del perfil rectangular. El resto de los diagramas pueden ser vistos en el Anexo.

[9] GRASP es un acrónimo que significa General Responsibility Asignment Software Patterns (patrones generales de software para asignar responsabilidades). El nombre se eligió para indicar la importancia de captar (grasping) estos principios, si se quiere diseñar eficazmente el software orientado a objetos.(26)

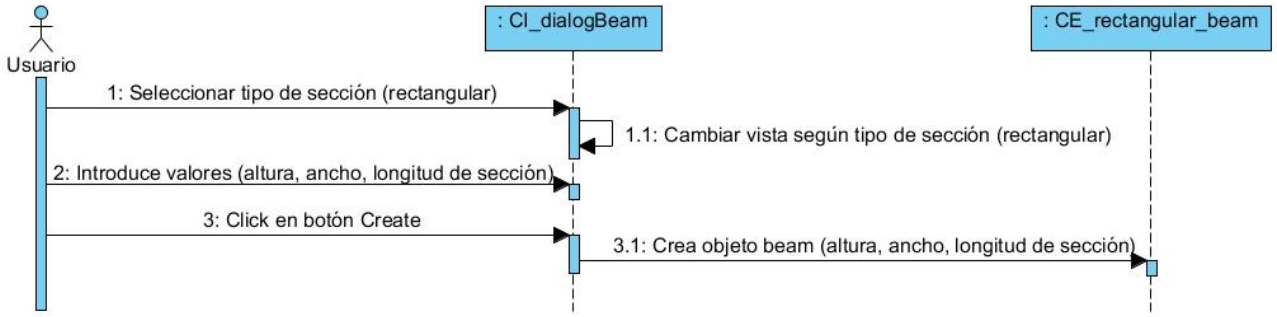


Figura 9 : Modelar la viga a partir del perfil rectangular.

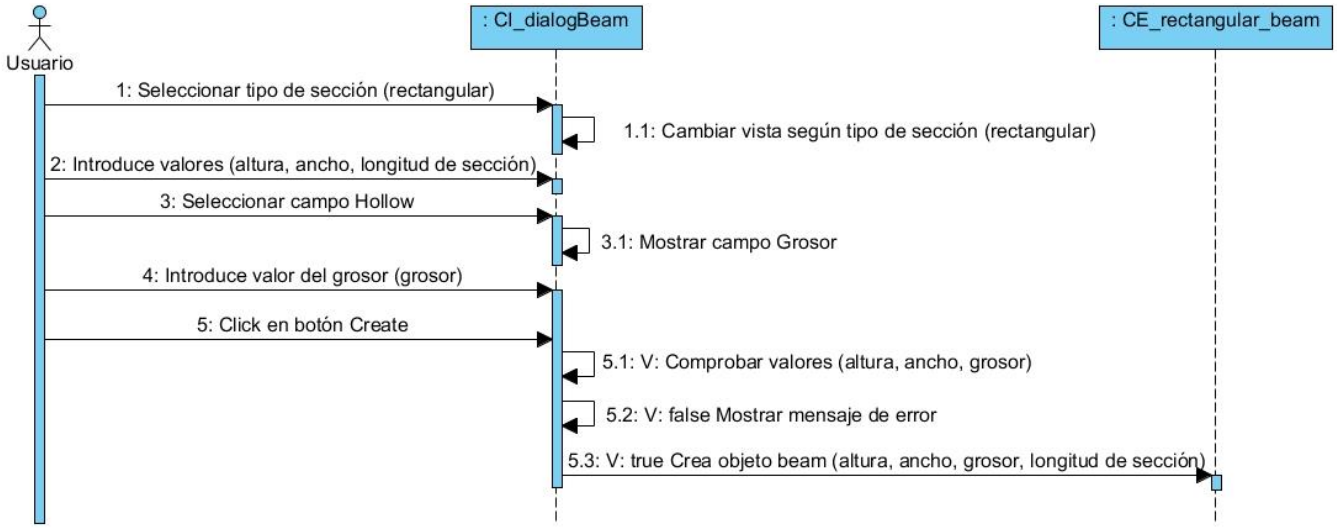


Figura 10 Modelar la viga a partir del perfil rectangular sección hueca.

2.5.5 Modelo de datos

Para que el componente propuesto sea capaz de modelar las vigas a partir de perfiles estructurales normalizados primeramente se debe almacenar la información respecto a sus propiedades físicas. Para lograr este propósito se diseña un modelo de datos relacional, tal como se observa en la Figura 11.

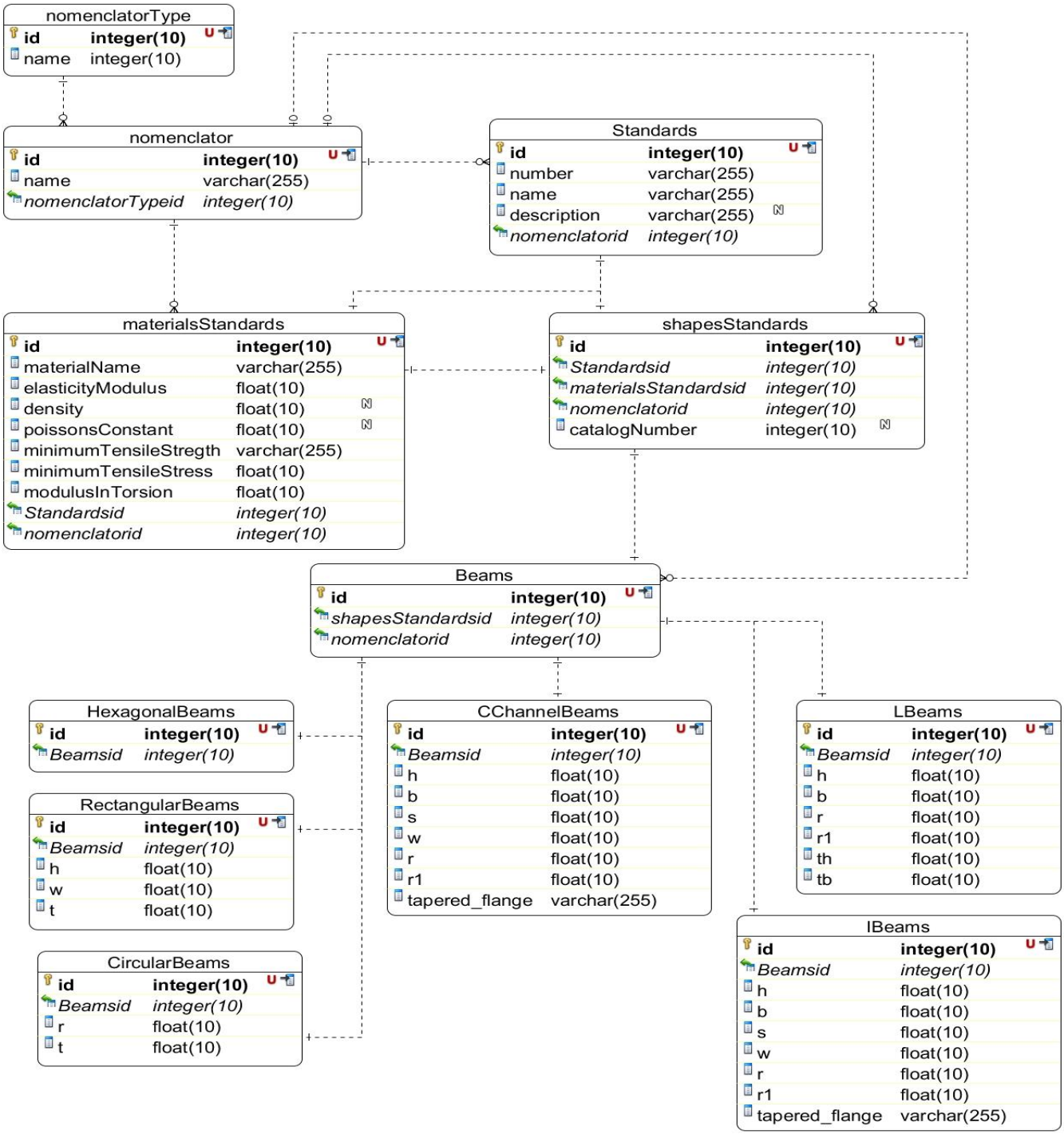


Figura 11 Modelo Entidad-Relación utilizado en la base de datos.

Descripción de las tablas de la base de datos.

A continuación, se muestra una breve descripción de los atributos contenidos en la tabla RectangularBeams, el resto de las descripciones ver en los Anexos.

Tabla 3 Descripción de la tabla RectangularBeams.

RectangularBeams		
Descripción: Almacena los parámetros de los perfiles de tipo rectangular de todas las normas almacenadas.		
Atributo	Tipo	Descripción
Id	integer(10)	Etiqueta única que identifica el perfil rectangular.
Beamsid	integer(10)	Almacena el identificador único de la tabla *Beams.*
h	float(10)	Almacena la altura del rectángulo.
w	float(10)	Almacena el ancho del rectángulo.
t	float(10)	Almacena el grosor del rectángulo en caso de que el perfil sea hueca.

2.6 Conclusiones parciales

En el presente capítulo se expuso:

- La descripción de la propuesta de solución para un mayor entendimiento de la misma
- El diseño del modelo del dominio para una mejor comprensión de los conceptos asociados.
- La definición de los requisitos funcionales agrupados en historias de usuarios con sus respectivos prototipos de interfaz, así como la de los requisitos no funcionales.
- El diseño del diagrama de clases que permite conocer la estructura y las relaciones de las clases que se manejan en dicho módulo.
- La descripción de los patrones de diseño definidos en la estructura arquitectónica de la solución que permiten el mantenimiento, entendimiento y reutilización de las clases.
- El diseño del modelo físico de datos, donde se almacenarán los datos de las normas que se gestionarán en el módulo.

Los artefactos obtenidos en este capítulo sientan las bases para la implementación de la solución propuesta.

3 Implementación y pruebas

En el presente capítulo se procede a realizar la implementación y las pruebas de software para garantizar la obtención de un producto de calidad. Se crea el diagrama de componentes del sistema y se definen los estándares de codificación empleados para el desarrollo de la solución. Son definidos los niveles y técnicas de pruebas que le serán aplicadas a la aplicación. Se diseñan también casos de pruebas que servirán para llevar a cabo estas comprobaciones de calidad.

3.1 Implementación

En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.(20)

3.1.1 Estándar de codificación

Los estándares de codificación se utilizan durante la fase de implementación de un software para darle una estructura al código facilitando su comprensión y mantenimiento.(42)

A continuación, se describen las pautas adoptadas para la implementación de la solución propuesta.(42)

Tabla 4 Estándar de codificación.

Definición de Objetos, Clases, funciones y atributos	
Descripción	**Ejemplo**

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.	class Foo{
	cuerpo de la clase
	}
	class FooFirst{
	cuerpo de la clase
	}

Siempre se declara para todas las clases implementadas su respectivo destructor de clase.	virtual ~Foo()
La declaración de funciones o métodos siempre comenzara en letra inicial minúscula. En caso de ser un nombre compuesto se	<Tipo dato retorno> funcion() <Tipo dato retorno> funcionCompuesta()

regirá por la normativa CamelCase-lowerCamelCase.	<Tipo dato retorno> funcionDobleCompuesta()
Los atributos siempre estarán escritos con letra minúscula. En caso de ser un nombre compuesto se regirá por la normativa CamelCase-lowerCamelCase.	<Tipo dato> atributo; <Tipo dato> atributoNombreCompuesto;
Definición de parámetros dentro de las funciones y constructores de clases	
Descripción	**Ejemplo**
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>)
Definición de expresiones	
Descripción	**Ejemplo**

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[];

Definición de estructuras de control y bucles	
Descripción	**Ejemplo**

Las estructuras de control y los bucles estarán definidos de igual manera en ambos casos siguiendo el estándar determinado por el *framework* 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
	}

Comentarios en el código según el estándar de C++	
Comentarios pequeños.	/* comentario sencillo */
Otros comentarios.	/* *Comentario */
Comentario de versión, descripción de clase y otras características de la clase o paquete.	/* ********** *Comentario amplio * ********** */

3.1.2 Diagrama de componentes

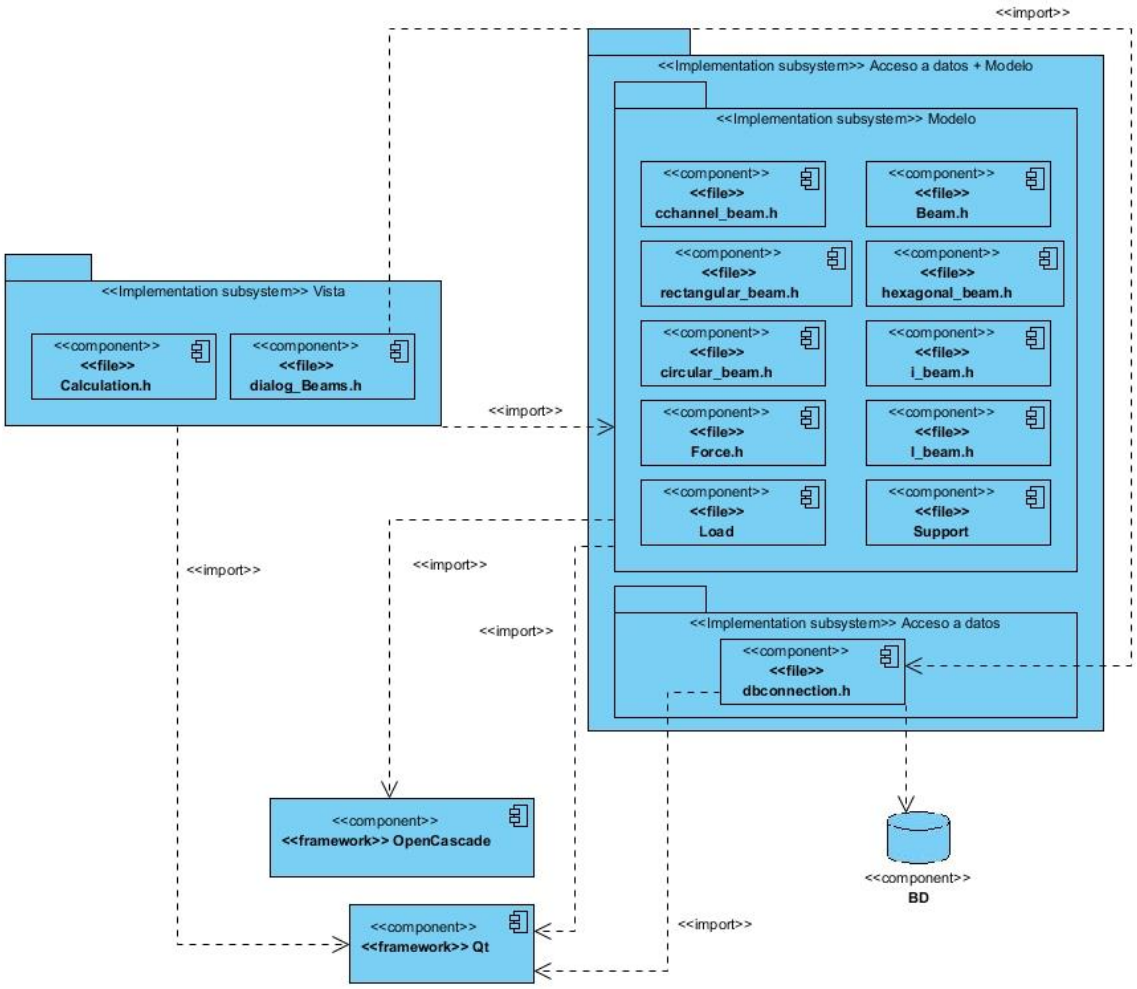


Figura 12 Diagrama de componentes

3.1.3 Creación de los perfiles estructurales

El proceso de construcción de perfiles estructurales se realiza teniendo en cuenta los parámetros introducidos por el usuario o extraídos de la base de datos que contiene las normas.

La tecnología *Open Cascade* cuenta con funcionalidades que permiten la confección de figuras primitivas como cilindros y cuadrados, además, de operaciones entre ellas que son utilizadas en la creación de perfiles de tipo rectangular, cuadrado y circular.

La tecnología *Open Cascade* define una jerarquía útil a la hora de crear entidades geométricas, la cual está compuesta por vértices, aristas, contorno, caras y sólidos, como se muestra en la Figura 13.

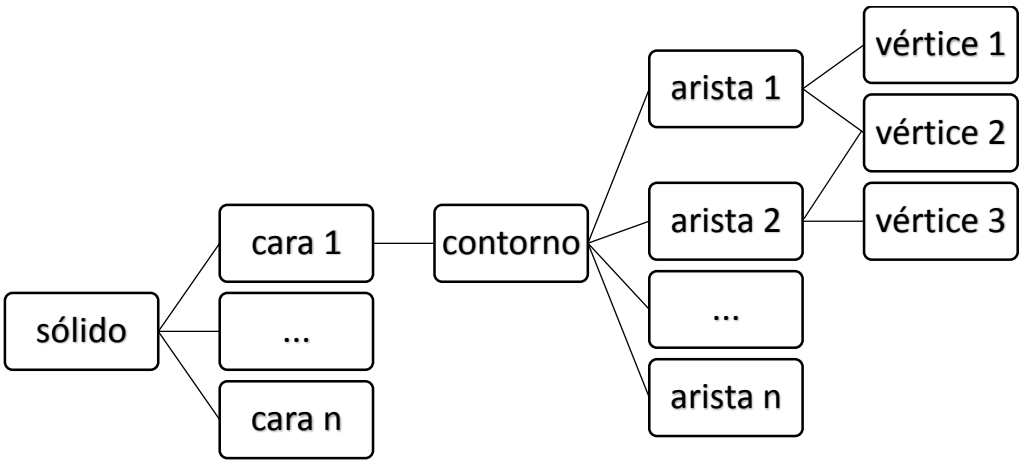


Figura 13 Jerarquía para la creación de entidades geométricas en Open Cascade.

Podemos hacer uso de esta jerarquía para crear, por ejemplo, un perfil de tipo I de alas rectas, para eso es necesario seguir un orden lógico de pasos como el mostrado en la Figura 14.

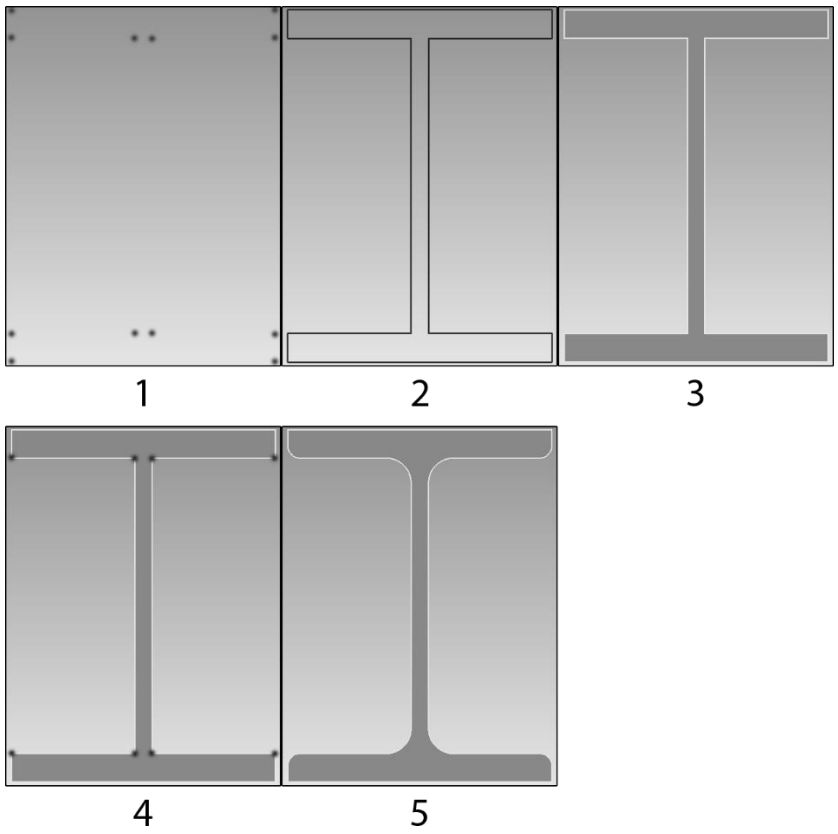


Figura 14 Pasos para crear una viga I de alas rectas.

Lo primero es definir el punto inicial en las coordenadas PA(x,y,z)=(0,0,0), luego se calcula la posición de cada uno de los demás puntos que conforman el perfil, por ejemplo, se crea el punto extremo inferior derecho (ver Figura 14 cuadro 1) con las coordenadas PB(x,y,z)=(***b***,0,0), donde ***b*** es la longitud de la base del perfil. De esta manera, utilizando los valores de los parámetros para conformar este tipo de perfil abordados en el epígrafe 1.2, se calculan las coordenadas del resto de los puntos.

Una vez calculadas las coordenadas de todos los puntos se crean las aristas uniendo los puntos consecutivos hasta que se obtiene un objeto semejante al mostrado en el cuadro 2 de la Figura 14. Luego todas las aristas se convierten en un contorno obteniendo un objeto semejante al mostrado en el cuadro 3 de la Figura 14. El contorno se transforma en una cara, que es el área dentro de dicho contorno, esta transformación es importante porque permite que la viga se cree como un sólido.

Una vez que se tenga la cara se procede a eliminar las esquinas del perfil que lo requieran, este paso es exclusivo cuando se confeccionan perfiles a partir de algún estándar. Para lograr este paso es necesario hacer un recorrido por la geometría de la cara del perfil para extraer todos los puntos y luego redondear los puntos según el radio de redondeado correspondiente a la norma.

Por último, se crea la viga teniendo en cuenta la cara del perfil y su longitud a partir del proceso de extrusión explicado en el epígrafe 1.2.

Para obtener vigas con las alas inclinadas solo se tiene en cuenta que el grosor de las alas está en la distancia $\frac{bf}{4}$ en caso del perfil I y en $\frac{\mathrm{bf}-\mathrm{tw}}{2}$ en caso del perfil C y con un ángulo de inclinación de $12{,}6°$ y $7{,}6°$ respectivamente. Para estos casos se calcula el valor de los nuevos puntos que le darán la inclinación al ala utilizando la ecuación trigonométrica de la tangente de un ángulo (ver Figura 15Figura 14 cuadro 1), el resto de los pasos se mantienen invariables.

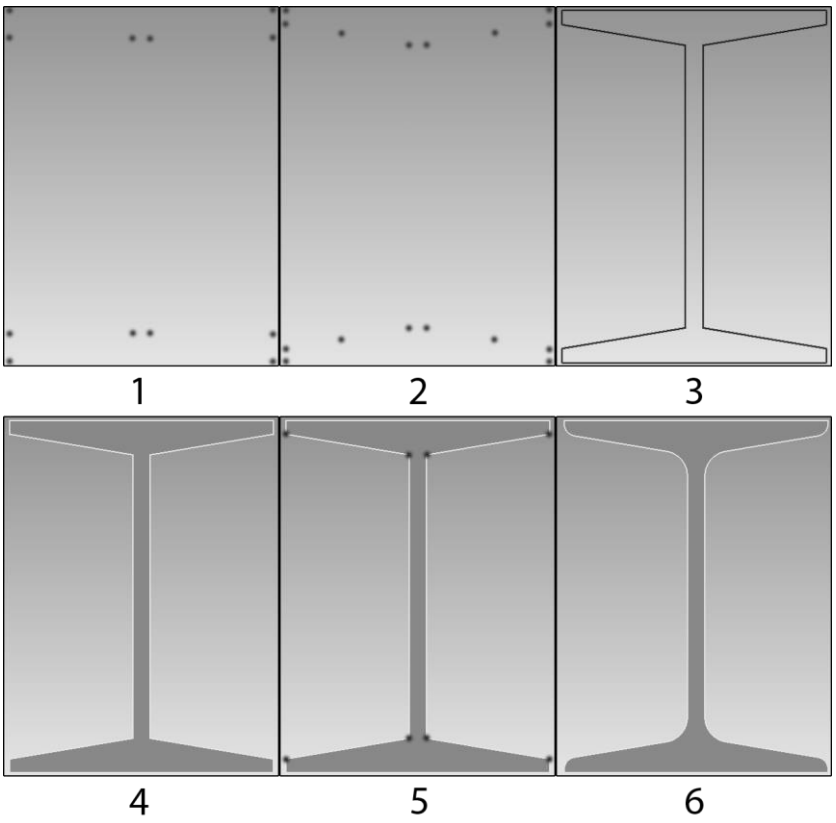


Figura 15 Pasos para crear una viga I de alas inclinadas.

De esta manera se obtienen las vigas de tipo I, L y C utilizando cualquiera de las variantes explicadas anteriormente dependiendo de la norma que se escoja o de los datos introducidos por el usuario.

3.2 Pruebas

El proceso de pruebas del software tiene 2 objetivos distintos (35):

- Para demostrar al desarrollador y al cliente que el software satisface sus requerimientos.
- Descubrir defectos en el software en el que el comportamiento de este es incorrecto, no deseable o no cumple su especificación.

3.2.1 Etapas definidas por AUP-UCI para el proceso de pruebas

La metodología AUP-UCI define 3 etapas de pruebas:

Pruebas internas: se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser

liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible componentes de pruebas ejecutables para automatizar las pruebas.(20)

Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.(20)

Pruebas de Aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.(20)

Para la presente investigación se tienen en cuenta las pruebas internas, específicamente las pruebas unitarias y las pruebas funcionales, y las pruebas de aceptación.

3.2.2 Pruebas internas

Para la realización de las pruebas internas se tienen en cuenta las 2 categorías de pruebas: las pruebas de caja blanca que son las que se centran en la estructura de control del programa y las pruebas de caja negra que se centran en validar los requisitos funcionales sin tener en cuenta el funcionamiento interno del programa.(43)

3.2.2.1 Pruebas unitarias

Las pruebas unitarias se encuentran dentro de las técnicas de pruebas de caja blanca pues se “concentran en la lógica del procesamiento interno y en las estructuras de datos dentro de los límites de un componente”(38).

Para realizar las pruebas unitarias se utiliza el framework Qtest. A continuación, se muestra una imagen que evidencia la realización de las pruebas unitarias sobre el componente y la aceptación de todas las comprobaciones.

Figura 16 Resultado de las pruebas unitarias realizadas con QTest

3.2.2.2 Pruebas funcionales

Para validar los requisitos funcionales de la aplicación se diseñan casos de prueba a partir de las historias de usuario, con el objetivo de mostrar la mayor cantidad posible de deficiencias en la aplicación.

A continuación se muestra

3.2.3 Resultados de las pruebas

Con la ejecución de las pruebas de caja negra se lograron identificar un grupo de no conformidades.

Conclusiones generales

Una vez finalizado el presente trabajo de diploma, se puede concluir que:

- La investigación realizada evidenció que no existe actualmente ningún componente que automatice el proceso de diseño en las aplicaciones CAD libres (FreeCAD y LibreCAD).
- El componente se puede ajustar a cualquier aplicación de código abierto basada en la tecnología *Open Cascade***.**

Bibliografía

1. Diccionario de informatica, internet, tecnologias y computacion on line. [online]. 2016. [Accessed 25 May 2016]. Available from: http://www.alegsa.com.ar/Diccionario/diccionario.php

2. Areas Técnicas. [online]. 5 February 2016. [Accessed 5 February 2016]. Available from: http://www.unizar.es/aeipro/finder/INGENIERIA%20DE%20PRODUCTOS/BF04..htm

3. MONTANO, José Luis Montes de Oca. La migración hacia software libre en Cuba: complejo conjunto de factores sociales y tecnológicos en el camino de la soberanía nacional. *Revista Universidad y Sociedad* [online]. December 2015. Vol. 7, p. 119–125. Available from: http://rus.ucf.edu.cu/

4. RODRÍGUEZ, Iván Ávila. *MÓDULO PARA CREACIÓN Y UNIÓN DE ELEMENTOS VIGA ESTANDARIZADOS.* La Habana, Cuba : Universidad de las Ciencias Informáticas, 2012.

5. ISO. Sitio Oficial ISO. *Sitio Oficial* [online]. [Accessed 20 February 2016]. Available from: http://www.iso.org/iso/home.html

6. INSTITUTO URUGUAYO DE NORMAS TÉCNICAS UNIT. UNIT - Beneficios de la normalización. [online]. [Accessed 24 May 2016]. Available from: http://www.unit.org.uy/normalizacion/beneficios/

7. NUCOR and -YAMATO STEEL COMPANY. *Structural Shapes*. November 2014.

8. Autodesk Inventor Professional 2017. [online]. 4 May 2016. [Accessed 4 May 2016]. Available from: http://www.imaginit.com/software/autodesk-products/inventor

9. MUNFORD, Paul and NORMAND, Paul. *Mastering Autodesk Inventor 2016 and Autodesk Inventor LT 2016: Autodesk Official Press*. John Wiley & Sons, 2015. ISBN 978-1-119-05998-1.

10. FreeCAD: An open-source parametric 3D CAD modeler. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: http://www.freecadweb.org/

11. Solid Edge: Siemens PLM Software. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: https://www.plm.automation.siemens.com/en_us/products/solid-edge/

12. CATIA. [online]. Available from: http://www.3ds.com/es/productos-y-servicios/catia/

13. MITCalc - Mechanical, Industrial and Technical Calculations. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: http://www.mitcalc.com/

14. eFunda: About Us. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: http://www.efunda.com/about/about.cfm

15. SKYCIV ENGINEERING. SkyCiv Engineering. [online]. 2015. [Accessed 24 May 2016]. Available from: https://skyciv.com

16. SKYCIV ENGINEERING. Bending Moment and Shear Force Diagram Calculator. [online]. [Accessed 24 May 2016]. Available from: http://bendingmomentdiagram.com/

17. Programa XVIGAS. [online]. 3 October 2007. [Accessed 24 May 2016]. Available from: http://xvigas.sourceforge.net/index.html

18. www.excelcalcs.com - Enhanced BEAMANAL. [online]. [Accessed 24 May 2016]. Available from: https://www.excelcalcs.com/repository/strength/beams/enhanced-beamanal/

19. EDEKI, Charles. Agile Unified Process. [online]. September 2013. Vol. Vol.1, p. pg. 1 3-1 7. [Accessed 26 May 2016]. Available from: http://www.ijcsma.com/publications/september2013/V1I304.pdf

20. SÁNCHEZ, Tamara Rodríguez. *Metodología de desarrollo para la Actividad productiva de la UCI.* La Habana, Cuba : Universidad de las Ciencias Informáticas, 2015.

21. Open CASCADE Technology: Overview. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: http://dev.opencascade.org/doc/overview/html/index.html#OCCT_OVW_SECTION_1

22. OPEN CASCADE COMMUNITY EDITION. Open CASCADE Technology, The Open Source 3D Modeling Libraries | Collaborative development portal. [online]. 2016. [Accessed 25 May 2016]. Available from: http://dev.opencascade.org/

23. About Qt - Qt Wiki. [online]. 2 November 2015. [Accessed 24 May 2016]. Available from: https://wiki.qt.io/About_Qt

24. BJARNE STROUSTRUP. *The C++ Programming Language. Third Edition*. 3ra. AT&T Labs Murray Hill, New Jersey : Addison-Wesley, 1997. ISBN 0-201-88954-4.

25. VISUAL PARADIM. Software Design Tools for Agile Teams, with UML, BPMN and More. [online]. [Accessed 25 May 2016]. Available from: https://www.visual-paradigm.com/

26. LARMAN, Craig. *UML y Patrones. Introducción al analisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999. ISBN 970-17-0261-1.

27. Sobre PostgreSQL. [online]. 5 May 2016. [Accessed 5 May 2016]. Available from: http://www.postgresql.org.es/sobre_postgresql

28. *Bases de datos en MySQL.pdf* [online]. [Accessed 5 May 2016]. Available from: http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02151.pdf

29. About SQLite. [online]. 18 January 2016. [Accessed 18 January 2016]. Available from: https://www.sqlite.org/about.html

30. DORNEZ, Walter. Perfiles Estructurales. [online]. 27 December 2012. [Accessed 18 May 2016]. Available from: https://prezi.com/tlj8pxaserht/perfiles-estructurales/

31. FERDINAND P. BEER, E. RUSELL JOHNSTON, JR. and ELLIOT R. EISENBERG. *MECÁNICA VECTORIAL. para INGENIEROS. Estática.* 8va. México, D.F : McGraw-Hill Interamericana, 2007. ISBN 978-970-6103-9.

32. soporte - Definición - WordReference.com. [online]. 2016. [Accessed 25 May 2016]. Available from: http://www.wordreference.com/definicion/soportesoporte - Significados en español y discusiones con el uso de “soporte”.

33. SÁNCHEZ, Rafael Sánchez. PRACTICA: Funcionamiento, descripción y esquematización de máquinas y mecanismos. - Tema 1 Conceptos fundamentales. [online]. Universidad de Huleva. July 2009. [Accessed 25 May 2016]. Available from: http://www.uhu.es/rafael.sanchez/disenodemaquinas/carpetadeapuntesdedisenodemaquinas/Tema%201%20Conceptos%20fundamentales.pdf

34. CAPITULO V. Modelación y Análisis Estructural de Edificios Altos. [online]. [Accessed 25 May 2016]. Available from: http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/474/A8.pdf?sequence=8

35. SOMMERVILLE, Ian. *Ingeniería de Software. Séptima edición*. 7ma. Madrid : PEARSON EDUCACIÓN, S.A, 2005. ISBN 84-7829-074-5.

36. LETELIER, Patricio and PENADÉS, Mª Carmen. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* Universidad Politécnica de Valencia, [no date].

37. IEEE COMPUTER SOCIETY. *Guide to the Software Engineering Body of Knowledge.* 2004.

38. PRESSMAN, Roger S. *Ingeniería de software. Un enfoque práctico.* [no date].

39. REYNOSO, Carlos and KICILLOF, Nicolás. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Universidad de Buenos Aires, 2004.

40. ARQHYS. La escalabilidad. [online]. [Accessed 25 May 2016]. Available from: http://www.arqhys.com/construcciones/escalabilidad.html

41. RUMBAUGH, James, JACOBSON, Ivar and BOOCH, Grady. *El Lenguaje Unificado de Modelado. Manual de referencia.* Addison-Wesley, 2000.

42. FERRER, Abel Fernández. *Componente para la modelación de árboles escalonados en el Sistema CAD 2D.* La Habana, Cuba : Universidad de las Ciencias Informáticas, 2015.

43. ROBERTO RUIZ TENORIO. *Las pruebas de software y su importancia en las organizaciones*. Xalapa-Enríquez, Veracruz : Facultad de Contaduría y Administración. Universidad Veracruzana, 2010.