Universidad de las Ciencias Informáticas

Facultad 5

Editor de estilos CSS para interfaces Ribbon de programas en Qt.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autora: Madonna Anguela Figueredo.

Tutor: M. Sc. Osvaldo Pereira Barzaga.

Co-tutor(es): Ing. Rubén Alcolea Núñez.

Ing. Luis Guillermo Silva Rojas.

La Habana, Marzo de 2013

DECLARACIÓN DE AUTORÍA:

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

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

Madonna Anguela Figueredo

Autora

M.Sc. Osvaldo Pereira Barzaga

Tutor

Ing. Rubén Alcolea Núñez

Cotutor

Ing. Luis Guillermo Silva Rojas

Cotutor

DATOS DE CONTACTO

Tutor: MSc. Osvaldo Pereira Barzaga.

Edad: 28

Ciudadanía: Cubano.

Institución: Instituto de Cibernética, Matemática y Física (ICIMAF).

Título: MSc. en Informática Aplicada.

Categoría Docente: Instructor.

Email: opereira@icimaf.cu

M.Sc. Osvaldo Pereira Barzaga. Graduado en el año 2008 como Ing. en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI). Con 8 años de experiencias en los temas de visualización médica y procesamiento digital de imágenes, desde el 2008-2009 se desempeña como líder del Proyecto Simulador Quirúrgico. En el 2009-2011 es el líder del área temática de visualización científica y líder del proyecto Vismedic. En el año 2010 se gradúa de máster en la disciplina de Informática Aplicada. Durante el 2012 cumple misión internacionalista en Venezuela, desempeñándose como arquitecto general del software GALBA-CAD, desarrollado por el proyecto CDSEM en el marco de la empresa mixta entre PDVSA[1] y Guardián del Alba. Actualmente posee la categoría docente de instructor y trabaja en el grupo de procesamiento de imágenes digitales del Dpto. de Matemática Disciplinaria en el Instituto de Cibernética, Matemática y Física (ICIMAF).

[1] PDVSA: Petróleos de Venezuela Sociedad Anónima.

Co-tutor: Ing. Luis Guillermo Silva Rojas.

Edad: 25

Ciudadanía: Cubano

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ing. Ciencias Informáticas.

Categoría Docente: Recién Graduado.en Adiestramiento.

Email: lgsilva@uci.cu

Graduado de la UCI, con cuatro años de experiencia en el tema de la Gráfica Computacional y líder de un proyecto de Realidad Virtual Vismedic en la Universidad de las Ciencias Informáticas.

Co-tutor: Ing. Rubén Alcolea Núñez.

Edad: 25

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ing. Ciencias Informáticas.

Categoría Docente: Recién Graduado.en Adiestramiento.

Email: ralcolea@uci.cu

Graduado como Ing. en Ciencias Informáticas en la UCI en el año 2011. Tiene cuatro años de experiencias en el procesamiento digital de imágenes y la visualización médica. Actualmente es arquitecto del proyecto Vismedic, perteneciente a la Línea Visualización Científica del centro CEDIN. Es miembro del Colectivo de Entrenadores del Movimiento ACM-ICPC en la UCI.

DEDICATORIA

A mi madre que además de darme la vida, ha estado siempre pendiente de mis luchas diarias, por su ejemplo, su dedicación y todo ese amor que me ha brindado en todos mis días aún sin tenerme a su lado,

…a mi padre por darme su apoyo incondicional cada vez que lo he necesitado, por brindarme de una forma u otra su amor y cariño, simplemente por estar ahí,

…a mi abuelita linda, la luz de mis ojos, una de las razones que me impulsa a seguir adelante en la vida,

…a mi hermanita Kiara, que aunque no esté a su lado, nunca sale de mis pensamientos.

Los AMO, gracias por ser mi luz aún en días de sol, por ser mis fuerzas y las razones por las que vivo y lucho en la vida.

AGRADECIMIENTOS

Aprovecho este espacio para dar las gracias a todos aquellos que de una forma u otra contribuyeron a la realización de este trabajo, a mi formación profesional y a ser cada día mejor persona.

A mis cotutores Guille y Rubén, por brindarme su dedicación y paciencia siempre que los necesité, por darme los buenos regaños de vez en cuando, por compartir además su amistad incondicional. Gracias Rubén por aconsejarme en muchos momentos y así ayudarme a tomar decisiones.

A mi tutor Osvaldo por ser mi guía, por brindarme su apoyo incondicional, por ser exigente, siempre pidiendo más de lo que podemos lograr, lo que hizo que llegáramos hasta aquí, por ser más que tutor, un buen amigo.

A los amigos con los que he compartido mis 5 años de la carrera, llegando a convertirse en parte de mi familia, sobre todo las chicas del 92105, a las que siempre les digo: ¨Como voy a extrañar esto¨.

A mi compañero Peña, por haberme ayudado tanto las veces que lo necesité, que fueron muchas!!!

A mis ¨ amiguis ¨, Amal, Yamilka y Nany, con las que he compartido mucho de mi historia, mis momentos de alegrías, de tristezas y disgustos, se convirtieron en mis hermanitas.

A mi profe ¨amigui¨ Yadira, por sus consejos cuando los necesitaba, por brindarme su amistad y apoyo.

A mi mamá por ser mi vida, mis ojos, mi sol, mi todo, por guiarme siempre por el camino correcto, por su ejemplo, por su amor, por ser la luz que me guía en cada paso…

A mis hermanas Kiara, Chavelis, Lianet y Laura por darme su amor y cariño, por las que siempre estaré dispuesta para lo que necesiten…

A mi padrastro, mi segundo papá, por quererme como una hija, por su apoyo constante y sus consejos que nunca faltan…

A mi abuela María por ser parte de mí, por su ayuda incondicional, su preocupación constante, su ternura, por educarme y adorarme en todo momento…

A mi papá por su apoyo, su amor, su cariño, su presencia que son las cosas más importantes para mí, nunca hubiese querido tener otro papá…

A mi madrastra, por ser tan especial, por darme su amor, cariño y atención, la persona en la que me he apoyado en muchos momentos…

A mi tía Liset, por apoyarme, por quererme, por ser más que mi tía una hermana…

A mi ¨mango¨ mi sobrinito Nehemías, por ser mi cosita linda, mi alegría cada vez que pienso en él…

A toda mi familia en general que me ha apoyado en todo momento…

A Julio, por demostrarme que hay cosas que pueden volver a sentirse y encontrarse, por sus consejos y comprensión, su apoyo incondicional, por cada minuto de amor y alegría, por estar en mi vida…

A mis amigos y compañeros de grupo desde primer año, por las fiestas, por la alegría compartida, por los momentos de apoyo...

A todos los profes que han contribuido en mi formación como profesional…

A todos los que en un momento u otro, formaron parte de mi historia y ocuparon un pedacito de mí...

GRACIAS!!!

RESUMEN

Las interfaces ribbon se han utilizado en los últimos años para diseñar nuevas interfaces gráficas en aplicaciones tan diversas como el Office o el reciente sistema operativo Windows 8. Para diseñar este tipo de interfaces se utilizan editores que permiten modificar el estilo y distribución de los componentes gráficos en la misma. El uso de los editores de interfaces ribbon minimiza el tiempo de construcción de las aplicaciones así como la complejidad y el esfuerzo en el diseño.

El presente trabajo propone un editor de estilos CSS para interfaces ribbon que permita modificar la apariencia visual de los componentes gráficos de una ribbon utilizando el *framework* Qt. El editor brinda la posibilidad de importar archivos de código CSS para modificar y exportar nuevos estilos que puedan ser cargados por otras aplicaciones.

Como resultado se obtuvo un editor de estilos CSS multiplataforma, que agiliza la creación de nuevos estilos para interfaces ribbon, desarrolladas con el *framework* Qt sin necesidad de implementarla a nivel de código fuente en C++ y CSS.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Durante el curso 2011 – 2012 miembros de la línea Visualización Científica de la Facultad 5 en la Universidad de las Ciencias Informáticas se vieron inmersos en el desarrollo de un sistema para el diseño de piezas mecánicas de taladros perforadores de petróleo como parte del proyecto CDSEM[2] firmado por la empresa cubana de capital mixto en Venezuela "G*uardián del Alba".*

Los sistemas más utilizados por los especialistas de la unidad de diseño de la industria China-Venezolana de Taladros (ICVT) eran *AutoCAD*, *Inventor* y *SolidWorks*. Estos cuentan en su interfaz gráfica de usuario con una barra de herramientas que presenta en su arquitectura de información una interfaz ribbon.

Se define una **r**ibbon[3] como una interfaz gráfica de usuario, compuesta de una banda (cintas) en la parte superior de una ventana, donde se exponen todas las funciones que puede realizar un programa en un solo lugar [1].

Uno de los retos principales que enfrentó el equipo de desarrollo CDSEM, durante la implementación del sistema GALBA-CAD fue garantizar que la interfaz de usuario del sistema propuesto tuviera la misma arquitectura de información que los sistemas propietarios mencionados anteriormente.

Para lograr el diseño de interfaces ribbon en Qt, el cual es el *framework* de desarrollo utilizado por los miembros del equipo CDSEM, se hizo necesario implementar una biblioteca de clases que contiene los componentes fundamentales de la interfaz gráfica. Esta interfaz presenta como deficiencia, que el diseño debe ser elaborado a nivel de código fuente en C++ y CSS[4], ya que el editor QtDesigner no está diseñado para realizar interfaces del tipo ribbon. Esto ocasiona atrasos en los plazos de entrega del proyecto y bajos niveles de reutilización en los diseños de interfaces ribbon para otros productos.

[2] CDSEM: Centro de Diseño y Simulación de Estructuras Mecánicas.
[3] Ribbon: Cinta de opciones.
[4] CSS: Hojas de Estilo en Cascada.

A partir de la situación problémica anterior, se propone el siguiente **problema de investigación:** ¿Cómo obtener un estilo de interfaz ribbon utilizando el *framework* Qt, sin necesidad de implementarlo a nivel de código fuente en C++ y CSS?

Se define como **objeto de estudio** las interfaces ribbon y se plantea como **objetivo general:** Desarrollar un editor de estilos CSS para componentes de interfaces ribbon utilizando el *framework* Qt.

El objeto de estudio se enmarca en el **campo de acción** editores de estilos CSS para interfaces ribbon. El presente trabajo defiende la siguiente idea: la implementación del editor de estilos CSS para interfaces ribbon propuesto, permitirá la obtención de un estilo de ribbon, sin necesidad de implementarlo a nivel de código fuente en C++ y CSS utilizando el *framework* Qt.

A continuación se mencionan un conjunto de tareas que permitirán dar cumplimento al objetivo formulado:

- ✓ Elaboración del marco teórico de la investigación a partir del estado del arte del tema.
- ✓ Caracterización del editor de interfaces ribbon QtitanRibbon para conocer los estándares de diseño que debe cumplir el editor propuesto.
- ✓ Caracterización del *framework* Qt con el objetivo de conocer los elementos necesarios para la implementación del editor de estilos para interfaces ribbon.
- ✓ Diseñar la interfaz gráfica de usuario de la aplicación.
- ✓ Caracterización de las propiedades de estilos CSS para modificar la apariencia visual del editor de estilos de interfaces propuesto.
- ✓ Definir el formato de fichero para exportar el código CSS a partir del editor de estilos de interfaces propuesto.
- ✓ Implementación del editor de estilos para interfaces ribbon en Qt.
- ✓ Validación del editor de estilos para interfaces ribbon propuesto sobre Qt.

Los métodos científicos que se utilizan para el estudio precedente de este trabajo son los siguientes:

Métodos Teóricos:

Histórico – Lógico: Para analizar la evolución, desarrollo actual y las nuevas tendencias de los editores de estilos para interfaces ribbon.

Analítico – Sintético: Para buscar lo que caracteriza y distingue a los editores de estilos para interfaces ribbon, su funcionamiento y utilidad.

Modelación: Permite hacer una reproducción simplificada de la realidad con el objetivo de definir un esquema de la herramienta a través de diagramas de clases, de flujo y de componentes.

Métodos Empíricos:

Observación: Para apreciar los estándares de diseño adoptados en los editores de estilos para interfaces ribbon y así determinar el diseño para el editor propuesto.

Pruebas: Para validar los resultados obtenidos a partir de la investigación realizada.

Para conocer brevemente el contenido de este trabajo a continuación se hace una pequeña descripción de cada uno de los capítulos.

Capítulo 1 "Fundamentación Teórica".

En este capítulo se hace un estudio de un grupo de conceptos asociados a la investigación, con el fin de llegar al componente visual de estudio definido en la misma. Además de analizar y definir las distintas características de los editores de interfaces ribbon y el *framework* Qt, así también de cada una de las propiedades de los componentes usados en el editor propuesto.

Capítulo 2 "Solución propuesta".

Se propone la solución técnica al problema planteado. Se especifica un prediseño a utilizar del estilo de ribbon que se presenta, así también las propiedades de estilos CSS correspondientes a cada uno de los componentes utilizados. Se aborda el formato en que se va a exportar el código generado, al mismo tiempo en que el usuario crea su estilo de ribbon. Se define y justifica la metodología, herramienta y lenguajes utilizados para el diseño e implementación del

editor propuesto. Se describe además el sistema, desde la perspectiva de la Ingeniería de Software, se procede al levantamiento de requisitos del sistema haciendo un análisis más exhaustivo del componente visual de estudio que se plantea. Finalmente se ofrece la propuesta de solución del problema planteado al inicio de la investigación.

Capítulo 3 "Implementación y validación del sistema".

En este capítulo se abordan los temas relacionados con la implementación del sistema, basado en lo especificado en los capítulos anteriores. Luego se realizan casos de prueba para validar los resultados alcanzados.

Finalmente se elaboró un glosario de términos con el objetivo de facilitar la comprensión del lenguaje usado. Un índice de figuras y otro de tablas para tener mejor organización de estos elementos, así como las referencias bibliográficas por si el lector siente la necesidad de ampliar sus conocimientos sobre el tema.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

En este capítulo se hace un estudio de los conceptos de interfaz ribbon, para qué se utilizan y por qué son útiles, qué es un editor de interfaz ribbon, tipos de editores de interfaces ribbon, además de mencionar las características y principios de diseño que tienen las mismas. Se profundiza en el *framework* de desarrollo Qt, las características del editor de interfaces gráficas del mismo, así como el funcionamiento y propiedades de cada uno de los elementos utilizados en la interfaz ribbon del editor propuesto. Se abunda en los conceptos de CSS, propiedades de estilos CSS y los tipos de editores de estilos CSS.

1.1 Interfaz ribbon

Se define una ribbon como una interfaz gráfica, que se ubica en la parte superior de una ventana, en ella se exponen todas las funciones que puede realizar un programa. Por lo que también se le conoce como cinta de funciones [2].

A partir de los conceptos analizados anteriormente, la autora expone a continuación el concepto de ribbon:

Ribbon, es una interfaz gráfica de usuario que se caracteriza por agrupar funciones de un programa en bandas accesibles a través de pestañas, en la parte superior de la ventana.

Fig. 1 Ejemplo de interfaz ribbon en Microsoft Office 2010.

Las interfaces ribbon [3] fueron creadas por la empresa Microsoft y se introdujeron por primera vez en el software Microsoft Office 2007.

El propósito de la ribbon es proporcionar un acceso rápido a las tareas más comunes dentro de cada programa. Por lo tanto, la ribbon se personaliza para cada aplicación y contiene comandos específicos para el programa. Además, la parte superior de la ribbon incluye varias fichas que se utilizan para revelar los diferentes grupos de comandos. Por ejemplo, la ribbon de Microsoft Word incluye Inicio, Insertar, Diseño de página, Referencias y otras fichas que muestran cada uno un conjunto diferente de comandos cuando se selecciona.

Existen 3 versiones de la interfaz ribbon, la primera salió en Office 2007, luego Microsoft creó una versión con una apariencia más simple y mejor rendimiento, denominada "*Scenic*" (Escena) y la incorporó en Paint, WordPad y algunos programas de Windows Live. Y posterior a eso, apareció una nueva ribbon en Office 2010, que contaba con la virtud de integrarse mejor con Aero[5].

Una ribbon típica está compuesta por grupos, que son un conjunto de comandos estrechamente relacionados. Además contienen:

- ✓ Contiene un menú de comandos que implican a hacer algo con un documento o área de trabajo, tales como archivos de comandos relacionados.
- ✓ Una barra de herramientas de acceso rápido, que es una pequeña barra personalizable que muestra los comandos más utilizados.
- ✓ Las fichas principales son las pestañas que siempre se muestran.
- ✓ Las fichas contextuales, aparecen sólo cuando un componente visual concreto está seleccionado. Las pestañas que siempre se muestran son llamadas pestañas centrales.
- ✓ Las fichas modales, que son fichas centrales muestran con un modo temporal determinado, como la vista previa de impresión.
- ✓ Las galerías, que son listas de comandos u opciones se presentan gráficamente. Una galería basada en los resultados ilustra el efecto de los comandos u opciones en lugar

[5] Aero: Interfaz gráfica utilizada en Windows Vista y Windows 7.

de los propios comandos. Una galería en la ribbon se muestra dentro de una ribbon, a diferencia de una ventana emergente.

- ✓ Información sobre herramientas mejoradas, que explican de manera concisa sus comandos asociados y dar las teclas de acceso directo. También pueden incluir gráficos y referencias para ayudar. Información sobre herramientas mejoradas para reducir la necesidad de comandos relacionados con ayuda.

- ✓ Lanzadores de cuadros de diálogo, que son botones en la parte inferior de algunos grupos que abren cuadros de diálogo que contienen funciones relacionadas con el grupo.

1.1.1 Aplicación y utilidad de las interfaces ribbon

La interfaz ribbon está pensada casi solo para software de edición (Word, Excel, Movie Maker, Paint) o de gestión más avanzada (Outlook), donde existe una gran cantidad de herramientas que deben ser organizadas de tal forma que sean accesibles con pocos clics, pero a la vez sean fáciles de descubrir. Están destinadas a mejorar la usabilidad, por la consolidación de las funciones del programa y los comandos en un lugar fácilmente reconocible, no es necesario mirar a través de múltiples niveles jerárquicos de los menús, las barras de herramientas o paneles de tareas antes de encontrar el comando, de ahí el porqué de su utilidad, además son la forma moderna para ayudar a los usuarios a encontrar, comprender y utilizar los comandos de forma eficaz y directa, con un número mínimo de clics, con menos necesidad de recurrir a la prueba y error, y sin tener que recurrir a la ayuda. Usando una ribbon aumenta el descubrimiento de características y funciones, permite el aprendizaje rápido del programa en su conjunto, y hace que los usuarios se sienten más en control de su experiencia con el programa.

En realidad, el objetivo principal de los ribbon reside en conglomerar las funcionalidades del programa en un solo lugar, a través de una especie de ventana emergente pequeña, con el fin de ofrecer facilidad de uso y rapidez para el usuario.

Por ello, los ribbon suelen ubicarse en la parte superior de la ventana o bien superpuestos a la ventana de la aplicación. Cabe destacar que en algunos programas se permite la personalización de su ribbon, brindando al usuario la posibilidad de elegir la ubicación y la apariencia del ribbon.

1.1.2 Editor de interfaz ribbon

Un editor de interfaz ribbon es una herramienta que permite a los desarrolladores crear su propia interfaz ribbon, para luego ser utilizada en otra aplicación.

A continuación se analiza algunas de las características del editor ***QtitanRibbon***. Este editor, es la base o modelo para la elaboración del editor de estilos CSS propuesto.

QtitanRibbon

QtitanRibbon [4] es la implementación de la interfaz de usuario ribbon de Microsoft para Qt. Es un sistema propietario que contiene una colección de widgets (elementos de control) que le ayudará rápidamente y fácilmente añadir una interfaz de usuario de próxima generación para su aplicación. El producto está completamente basado en Qt SDK[6].

El componente tiene un aspecto moderno para las tres principales plataformas que existen en el mundo, Windows, Linux y Mac OS X.

QtitanRibbon ofrece 5 temas para la aplicación final. Azul, Negro, Aqua, Plata y escénico tanto en la plataforma Windows, Linux y MAC OS X. Ejemplo de una captura de pantalla en Windows:

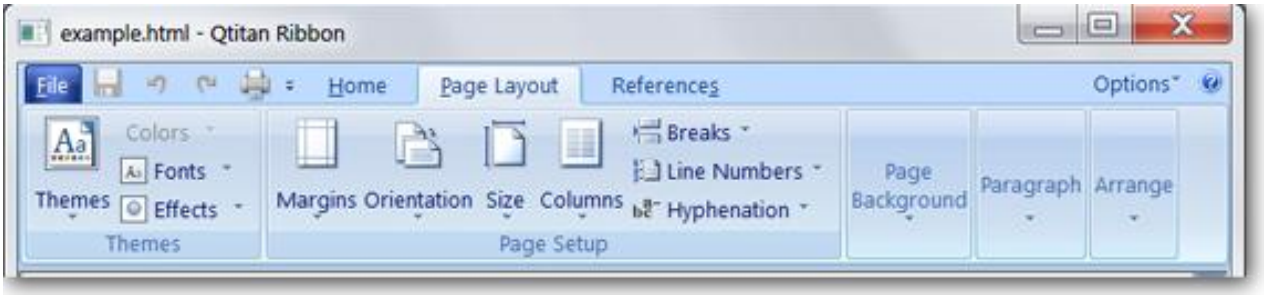


Fig. 2 Ribbon del programa QtitanRibbon en Windows

El componente está diseñado en un 100% nativo Qt / C++ y no utiliza los préstamos externos y bibliotecas de terceros. Sin embargo, hay porciones de código que se aplican en cada plataforma de diferentes maneras. Esto se hace debido a algunas características de Qt, por razón de mejorar el rendimiento del renderizado.

[6] SDK: Kit de Desarrollo de Software.

QtitanRibbon tiene una cómoda integración con QtDesigner. Esto le permite gestionar la ubicación de la interfaz de usuario de la ribbon en el formulario en el tiempo de diseño.

A partir de lo analizado anteriormente se propone tomar como patrón para la elaboración del editor de estilos CSS propuesto, el editor QtitanRibbon, pues su finalidad es lograr rápidamente y fácilmente una interfaz de usuario para una aplicación, solo que es un sistema propietario y resulta muy costoso para su uso en el centro.

1.1.3 Características de las interfaces ribbon

La interfaz gráfica que poseen los ribbon de una aplicación determinada se basa en una especie de barra de herramientas que cuenta con una serie de representaciones gráficas de los elementos de gestión de la aplicación, lo que además se encuentran agrupados por su tipo de funcionalidad.

Muchos de estos ribbon poseen un formato contextual, lo que hace que sólo aparezcan cuando es seleccionado un determinado componente visual, ofreciendo herramientas específicas para ello.

Además de las características mencionadas anteriormente están las siguientes:

- ✓ Facilitar que las personas encuentren y utilicen toda la gama de características que proporcionan las aplicaciones con interfaz ribbon.
- ✓ Agilizar el proceso de aplicar diseños, editar y cambiar el formato de forma que los usuarios puedan obtener excelentes resultados con menos tiempo y esfuerzo.
- ✓ Mantener un área de trabajo despejada que reduzca las distracciones de los usuarios y, de esa manera, que puedan dedicar más tiempo y energía a su trabajo.
- ✓ Lograr la simplicidad a la hora de buscar las características a utilizar en el programa.

1.2 Framework Qt

Qt es un *framework* multiplataforma para desarrollar interfaces gráficas de usuario [5]. Incluye una biblioteca y un IDE multiplataforma, además de herramientas de desarrollo integradas.

Qt es en la actualidad un *framework* de propósito general, comparable con *Swing* de *Java* o *.NET* de *Microsoft*, además ofrece una *suite* de aplicaciones para facilitar y agilizar las tareas de desarrollo. Las aplicaciones que componen esta *suite* son:

- *Qt Assistant*: Herramienta para visualizar la documentación oficial de Qt.
- *Qt Designer*: Herramienta WYSIWYG para crear interfaces de usuario.
- *Qt Linguist*: Herramienta para la traducción de aplicaciones.
- *Qt Creator:* IDE para el lenguaje C++, pero especialmente diseñado para Qt, integra las primeras dos herramientas mencionadas.

Qt es utilizado por empresas como *Intel*, *Google* o *Dreamworks.* Algunos ejemplos de aplicaciones desarrolladas utilizando Qt son:

- El entorno de escritorio *KDE.*
- *Google Earth.*
- *Skype.*
- *Virtual Box.*

El poder de este *framework* de desarrollo de software, está en sus bibliotecas que abarcan una gran variedad de tareas, la gestión de las bases de datos, creación de interfaces gráficas de usuario, el manejo de *sockets*, soporte para programación multihilo, comunicación con bases de datos, procesamiento de cadenas de caracteres, creación de gráficos 3D, entre otros. Para empezar a trabajar con él, se puede descargar gratuitamente el SDK de Qt, que incluye todas las herramientas para el desarrollo de software con ese *framework*: un IDE llamado *Qt Creator*, un compilador (MinGW para Windows y gcc para Linux), y un diseñador de interfaces llamado

Qt Designer, así como otras herramientas opcionales como *Qt Linguist* para crear aplicaciones multilenguaje y *Qt Assistant* que posee la documentación completa de las bibliotecas de Qt [5].

1.2.1 Editor avanzado de código

Qt Creator se centra en proporcionar características que ayudan a los nuevos usuarios de Qt a aprender y comenzar a desarrollar rápidamente, también aumenta la productividad de los desarrolladores con experiencia en Qt. Esta herramienta cuenta con:

- Editor de código con soporte para C++, QML y *ECMAscript.*
- Herramientas para la rápida navegación del código.
- Resaltado de sintaxis y auto-completado de código.
- Control estático de código y estilo a medida que se escribe.
- Soporte para *refactoring* de código.
- Ayuda sensitiva al contexto.
- Plegado de código (*code folding*).
- Paréntesis coincidentes y modos de selección.

1.2.2 Diseñador de Interfaces Gráficas de Usuario

El diseñador de interfaces gráficas de usuario es un entorno integrado para la creación y diseño de *forms* para proyectos C++ que permite diseñar rápidamente *widgets* y diálogos usando los mismos *widgets* que se usarán en su aplicación. Los *forms* son totalmente funcionales y pueden ser pre visualizado inmediatamente para asegurarse de que se verá y sentirá exactamente como el usuario lo pensó [6].

Con Qt se pueden desarrollar grandes aplicaciones gráficas, incluye soporte de nuevas tecnologías como OpenGL, XML, Bases de Datos, programación para redes, internacionalización y mucho más.

Qt dispone de una amplia gama de herramientas que facilitan la creación de formularios, botones y ventanas de diálogo con el uso del ratón.

Qt dispone de tres grandes ventajas:

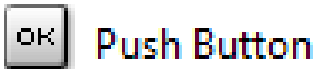
- ✓ Qt es completamente gratuito para aplicaciones de código abierto.
- ✓ Las herramientas, bibliotecas y clases están disponibles para casi todas las plataformas Unix y sus derivados (como Linux, MacOS X, Solaris.) como también para Windows, por lo que una aplicación puede ser compilada y utilizada en cualquier plataforma sin necesidad de cambiar el código y la aplicación se verán y actuará mejor que una aplicación nativa.
- ✓ Contiene una extensa biblioteca con clases y herramientas para la creación de buenas aplicaciones [5].

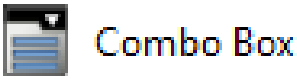
1.2.3 Componentes de Qt

Qt Designer cuenta con un panel de componentes que brinda una serie de widgets para mayor facilidad a la hora del diseño de interfaces gráficas o trabajo con ellas [7]. El editor de interfaces ribbon propuesto contendrá algunos de estos componentes que nos brinda el Designer, fundamentales para el trabajo con interfaces ribbon, estos se muestran a continuación:

Tabla 1 Componentes de Qt.

Componentes		Descripción	Vista previa
Buttons	**QToolButton**	Proporciona un componente de acceso rápido a los comandos u opciones que normalmente se utiliza dentro de un	 Tool Button

		QToolBar.	
	QRadioButton	Proporciona un componente de radio con una etiqueta de texto.	
	QPushButton	Es tal vez el widget más utilizado en cualquier interfaz gráfica de usuario. Se basa en presionar (clic) en un botón para manejar la computadora para realizar alguna acción, o para responder a una pregunta. Botones típicos como: Aplicar, Cancelar, Cierre, Sí, No y Ayuda.	
Containers	**QGroupBox**	Proporciona un marco de cuadro de grupo con un título.	
	QFrame	La clase ***QFrame*** es la clase base de widgets que puede	

		tener un marco.	
Widgets	**QComboBox**	Proporciona un medio para la presentación de una lista de opciones para el usuario de una manera que ocupa el mínimo de espacio en la pantalla.	
	QLineEdit	Es un editor de texto de una línea.	
	QSpinBox	Proporciona un widget cuadro de número.	
	QLabel	Ofrece una pantalla de texto o imagen.	
	QProgressBar	Ofrece una barra de progreso horizontal o vertical.	

1.3 Hojas de Estilo en Cascada (CSS)

CSS son las siglas de "*Cascade StyleSheet*" (Hojas de Estilo en Cascada).

Hojas de estilo en Qt

Las hojas de estilo en Qt son un mecanismo útil para personalizar la apariencia de los widgets, además de la clase ***QStyle***. Los conceptos, terminología y sintaxis de las hojas de estilo de Qt se basan en las hojas de estilo en cascada (CSS) para el lenguaje HTML, pero se adaptan para modificar el diseño de los widgets en Qt [7].

Las hojas de estilos son las especificaciones textuales que se pueden establecer en toda la aplicación utilizando ***QApplication::setStyleSheet()*** o en un widget específico con ***QWidget::setStyleSheet()***. Si varias hojas de estilos se encuentran en diferentes niveles, Qt determina cuál de estas hojas de estilos se aplica de todas las especificadas. Esto se llama en cascada.

Por ejemplo, la siguiente hoja de estilo especifica que todos los ***QLineEdit*** deben usar amarillo como color de fondo y todos los ***QCheckBox*** deben utilizar el rojo como color del texto:

```
QLineEdit {background: yellow}
 QCheckBox {color: red}
```

Para este tipo de personalización, las hojas de estilo son mucho más precisas que usando la clase ***QPalette***. Por ejemplo, los diseños del botón en rojo para una clase ***QPushButton*** usando la clase ***QPalette*** no se garantiza que funcione para las diferentes plataformas (Windows XP, Linux y Mac OS X), sin embargo usando las hojas de estilos se pueden visualizar los diseños en todas estas.

El uso de CSS permiten realizar todo tipo de personalizaciones que son difíciles o imposibles de realizar usando la clase ***QPalette***, por ejemplo, si establece el color de fondo rojo de un ***QPushButton***, puede estar seguro de que el botón no tendrá un fondo rojo en todas las plataformas. Es más recomendable utilizar las hojas de estilos para lograr fondos amarillos de los campos, texto rojo para pulsadores o casillas de verificación.

Las hojas de estilos se pueden utilizar para proporcionar un aspecto distintivo para su aplicación, sin tener que usar la clase ***QStyle***. Por ejemplo, puede especificar las imágenes arbitrarias para los botones de radio y casillas de verificación para hacer que se destaquen. Usando esta técnica, también se puede lograr personalizaciones menores que normalmente requieren varias subclases de las clases de estilos, como especificar un toque de estilo. A continuación se muestra un ejemplo de dos hojas distintivas:

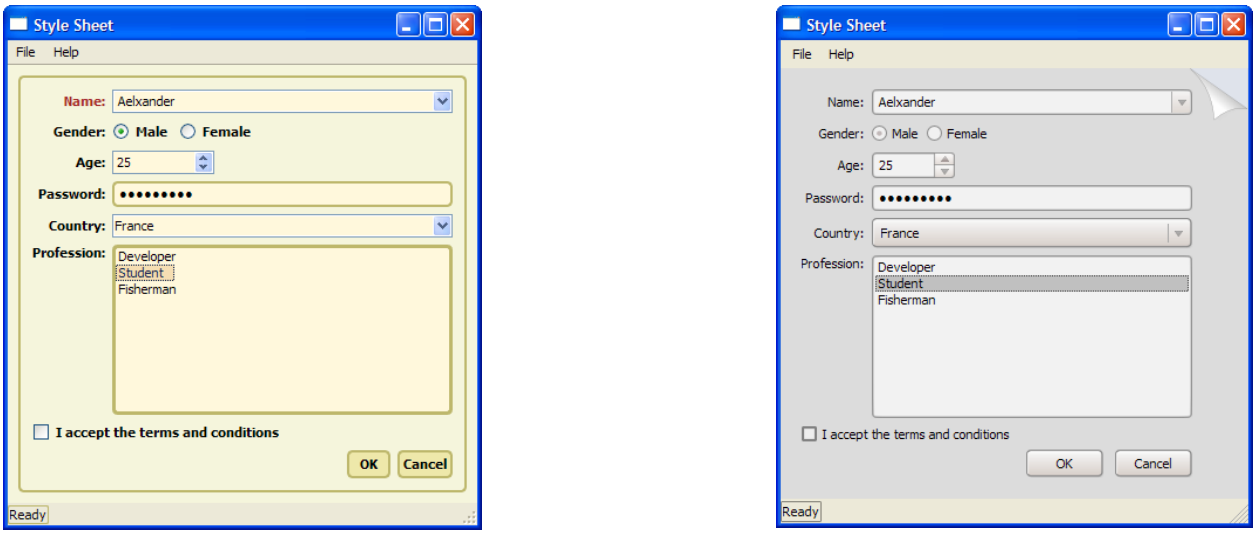


Fig. 3 Imágenes de la herramienta Assistant del Framework Qt.

Cuando una hoja de estilo está activa, el ***QStyle*** devuelve ***QWidget::estilo()***, es una envoltura de "hojas de estilo", no el estilo específico de la plataforma. El estilo envoltorio garantiza que se respete cualquier hoja de estilo activa y de otra manera continuar con las operaciones de dibujo en el estilo específico de la plataforma subyacente (por ejemplo, ***QWindowsXPStyle*** en Windows XP).

Se muestra un ejemplo de un programa simple:

```
QProgressBar {
     border: 2px solid grey;
     border-radius: 5px;
     text-align: center;
```

```
}
```

Las reglas mostradas anteriormente modifican el ancho del borde del ProgressBar a dos píxeles y define un radio de cinco píxeles, colocando el texto en el centro de la barra de progreso.

```
QProgressBar::chunk {
     background-color: #CD96CD;
     width: 10px;
     margin: 0.5px;
}
```

Este último código modifica el color de fondo de los pedazos que incluye la barra de progreso, ancho y margen que van a tener los mismos. A continuación se muestra el resultado de aplicar estas reglas:

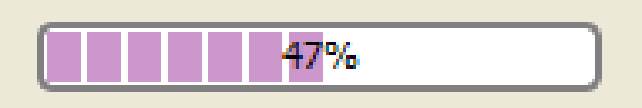


Fig. 4 Imagen de la herramienta Assistant del Framework Qt.

1.3.1 Propiedades de estilos CSS

Las propiedades de estilos son útiles para modificar desde la posición precisa de los elementos hasta el diseño de fuentes y estilos concretos. A continuación se muestra una tabla que resume algunas de las principales propiedades de estilos CSS:

Tabla 2 Propiedades de estilos CSS

Nombre del atributo	Posibles valores	Ejemplos
FUENTES-FONT		
Color	Valor RGB o nombre de color.	color: #009900; color: red;
Sirve para indicar el color del texto. Lo admiten casi todas las etiquetas de HTML. No todos los nombres de colores son admitidos en el estándar, es aconsejable entonces utilizar el valor RGB.		
Font-size	xx-small | x-small | small | medium | large | x-large |xx-large Unidades de CSS	font-size:12pt; font-size: x-large;
Sirve para indicar el tamaño de las fuentes de manera más rígida y con mayor exactitud.		
Font-family	serif | sans-serif | cursive | fantasy | monospace Todas las fuentes habituales.	font-family:arial,helvetica; font-family: fantasy;
Con este atributo indicamos la familia de tipografía del texto. Los primeros valores son genéricos, es decir, los exploradores las comprenden y utilizan las fuentes que el usuario tenga en su sistema. También se pueden definir con tipografías normales, como ocurría en HTML. Si el nombre de una fuente tiene espacios se utilizan comillas para que se entienda bien.		

Font-weight	normal | bold | border | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900	font-weight:bold; font-weight: 200;
Sirve para definir el ancho de los caracteres, o dicho de otra manera, para poner negrillas con total libertad. Normal y 400 es el mismo valor, así como bold y 700.		
Font-style	normal | italic | oblique	font-style:normal; font-style: italic;
Es el estilo de la fuente, que puede ser normal, itálica u oblícua. El estilo *oblique* es similar al *italic*.		
PÁRRAFOS-TEXT		
Text-align	left | right | center | justify	text-align: right; text-align: center;
Sirve para indicar la alineación del texto. Es interesante destacar que las hojas de estilo permiten el justificado de texto, aunque no tiene por qué funcionar en todos los sistemas.		
FONDO - BACKGROUND		
Background-color	Un color, con su nombre o su valor RGB	background-color: green; background-color: #000055;
Sirve para indicar el color de fondo de un elemento de la página.		

Background-image	El nombre de la imagen con su camino relativo o absoluto.	background-image: url(mármol.gif) ; background-image: url(http://www.x.com/fondo.gif)
Colocamos con este atributo una imagen de fondo en cualquier elemento de la página.		
BOX-CAJA		
Margin-left	Unidades CSS	margin-left: 1cm; margin-left: 0,5in;
Indicamos con este atributo el tamaño del margen a la izquierda.		
Margin-right		
Se utiliza para definir el tamaño del margen a la derecha.		
Margin-top	Unidades CSS	margin-top: 0px; margin-top: 10px;
Indicamos con este atributo el tamaño del margen arriba de la página.		
Margin-bottom	Unidades CSS	margin-bottom: 0pt; margin-top: 1px;
Con él se indica el tamaño del margen en la parte de abajo de la página.		

Padding-left	Unidades CSS	padding-left: 0.5in; padding-left: 1px;
Indica el espacio insertado, por la izquierda, entre el borde del elemento y el contenido de este. Es parecido a el atributo *cellpadding* de las tablas. El espacio insertado tiene el mismo fondo que el fondo del elemento.		
Padding-right	Unidades CSS	padding-right: 0.5cm; padding-right: 1pt;
Indica el espacio insertado, en este caso por la derecha, entre el borde del elemento y el contenido de este. Es parecido a el atributo *cellpadding* de las tablas. El espacio insertado tiene el mismo fondo que el fondo del elemento.		
Padding-top	Unidades CSS	padding-top: 10pt; padding-top: 5px;
Indica el espacio insertado, por arriba, entre el borde del elemento y el contenido de este.		
Padding-bottom	Unidades CSS	padding-right: 0.5cm; padding-right: 1pt;
Indica el espacio insertado, en este caso por abajo, entre el borde del elemento y el contenido de este.		
Border-color	color RGB y nombre de color	border-color: red; border-color: #ffccff;

Para indicar el color del borde del elemento de la página al que se lo aplicamos. Se puede poner colores por separado con los atributos *border-top-color*, *border-right-color*, *border-bottom-color*, *border-left-color*.		
Border-style	none | dotted | solid | double | groove | ridge | inset |outset	border-style: solid; border-style: double;
El estilo del borde, los valores significan: *none*=ningún borde, *dotted*=punteado (no parece funcionar), *solid*=solido, *Double*=doble borde, y desde *groove* hasta *outset* son bordes con varios efectos 3D.		
Border-width	Unidades CSS	border-width: 10px; border-width: 0.5in;
El tamaño del borde del elemento al que lo aplicamos.		

1.3.2 Editores de estilos CSS

JustStyle CSS Editor

Es un editor fácil de usar, se le llama software para los web masters. Es un estilo de entorno de desarrollo especializado de las hojas, con todo lo necesario para desarrollar y desplegar hojas de estilo en cascada. Ofrece una interfaz de usuario interactiva y presenta propiedades CSS en los asistentes especiales y listas genéricas de valor [8].

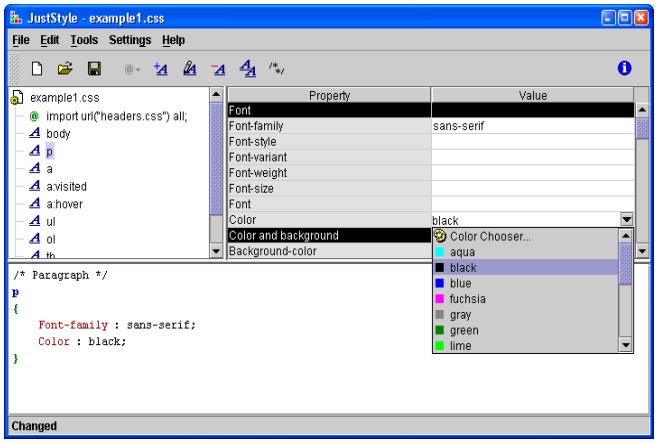


Fig. 5 Interfaz gráfica del editor de estilos CSS JustStyle.

JustStyle CSS Editor es una aplicación multiplataforma, escrito completamente en Java, funciona en diversas plataformas, como Microsoft Windows, IBM OS / 2, Linux, Apple Mac OS, Mac OS X y otros.

Simple CSS

La herramienta permite crear fácilmente hojas de estilo en cascada a partir de cero, y modificar los ya existentes. Con CSS simple, puede gestionar múltiples proyectos CSS e importar hojas de estilo existentes [9].

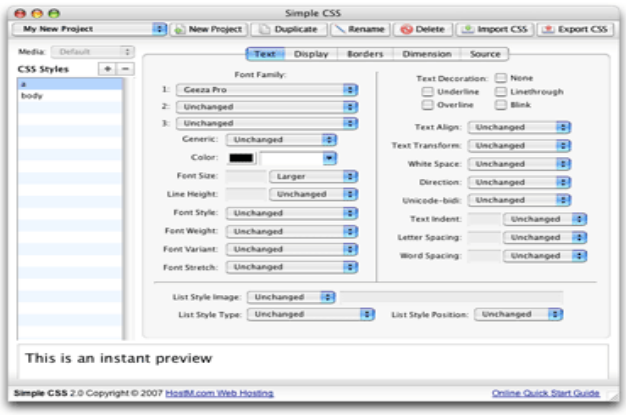


Fig. 6 Interfaz gráfica del editor Simple CSS.

QtQuick

QtQuick significa "Kit de Interfaz de Usuario" creado por Nokia junto al *framework* Qt y está compuesto por el lenguaje QML, nuevas herramientas de diseño integradas en *Qt Creator* y un *runtime* que da acceso a *API's* nativas o permite integrar QML y C++ [10].

Los principales usos para *Qt Quick* son la creación de interfaces, la rápida creación de prototipos y la creación completa de aplicaciones sencillas.

Las primeras versiones de *Qt Quick* sólo incluían primitivas (rectángulos, imágenes) orientadas a las interfaces de usuario táctiles. En nuevas versiones se están liberando conjuntos de componentes para cada plataforma soportada por Qt, lo cual incluye computadoras de escritorio, en las cuales no es tan común el uso de pantallas táctiles y por lo tanto el conjunto de componentes para escritorio podría no utilizar las capacidades táctiles de *Qt Quick* pero sí los beneficios de separación de lógica e interfaz y la velocidad de creación de interfaces.

Qt Quick es la propuesta de Qt para un nuevo enfoque sobre el desarrollo de aplicaciones e interfaces de usuario, donde se utiliza un lenguaje para la creación de estas interfaces y otro para la lógica de la aplicación, todo esto se hace mediante QML para la interfaz y *Javascript* o C++ para la lógica, en teoría esto permite una mejor organización del código y proporciona herramientas más adecuadas para la creación de interfaces, ya que los lenguajes utilizados para crearlas no son imperativos como era antes (el mismo tipo de lenguaje que para la lógica), si no lenguajes de marcado o declarativos.

Existe mucho código que utiliza los widgets tradicionales de Qt, por tanto se le sigue dando cierto soporte, aunque ya se ha dicho que a partir de Qt 5, Qt Quick será el kit de trabajo principal para Qt.

Qt Quick está compuesto por:

- ✓ Un sencillo lenguaje tipo scripting llamado QML (*Qt Meta-Object Language*).
- ✓ Herramientas en el IDE Qt Creator, incluyendo un editor visual que permite a los desarrolladores y diseñadores combinar elementos simples para construir interfaces gráficas de usuario animadas y fluidas.

- ✓ Un nuevo enfoque de programación declarativa, el cual hace más fácil crear interfaces gráficas de usuario diciéndoles que hacer, más que cómo hacerlo.

CAPÍTULO 2: SOLUCIÓN PROPUESTA

En el presente capítulo se propone una solución técnica para el desarrollo del editor de estilos para interfaces ribbon, partiendo de la fundamentación teórica que se ha planteado anteriormente. Se especifica el diseño de la ribbon propuesta, además de las propiedades de estilos CSS de cada uno de los componentes que tiene la misma. Se define el formato en que se va a exportar el código generado en tiempo real además de la metodología, herramientas y lenguajes utilizados para la implementación del editor propuesto.

Se presenta el modelo del dominio del problema, los requisitos funcionales y no funcionales. Se describen además los casos de usos obtenidos a partir de la captura de requisitos para brindar una mejor interpretación de las funcionalidades del sistema.

2.1 Editor de estilos CSS para interfaces ribbon

A partir del marco teórico elaborado en el capítulo 1, se propone el desarrollo de un editor de estilos CSS para interfaces ribbon, logrando la obtención de un nuevo estilo de ribbon que contiene componentes del *framework* Qt, sin necesidad de implementarla a nivel de código fuente.

El diseño de la interfaz del editor propuesto es una parte fundamental dentro del proceso de desarrollo del mismo, pues es la parte del sistema con que el usuario interactúa y la que le facilita además el acceso a las funcionalidades.

Se elaboró un diseño de interfaz con el propósito de que los usuarios se orienten con mayor facilidad y así darle solución a los problemas de accesibilidad y usabilidad, aspectos que se deben tener en cuenta a la hora de diseñar una interfaz gráfica de usuario. En la figura 7, 8 y 9 se muestran el diseño de interfaz del editor propuesto:

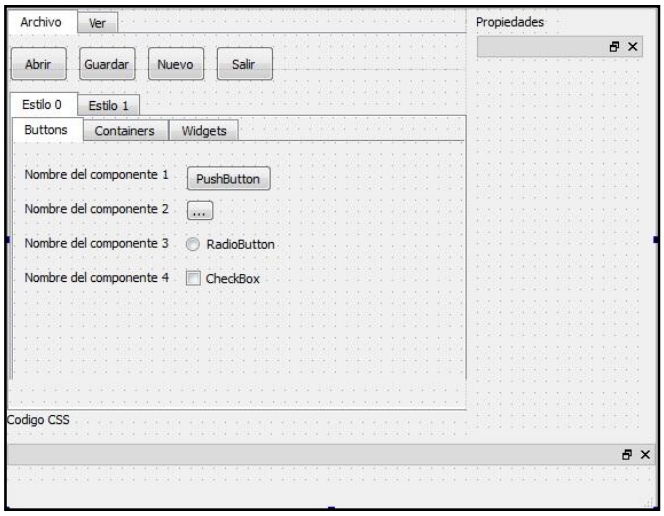


Fig. 7 Diseño de interfaz del editor propuesto. Tab Buttons.

2.2 Funcionalidades

El editor cuenta con un conjunto de funcionalidades, que se utilizan para mostrar, modificar y actualizar las propiedades de los componentes de la ribbon, así mismo para abrir, guardar, cambiarle el nombre a un estilo o simplemente añadir un nuevo estilo en una misma área de trabajo. A continuación se mencionan las funcionalidades esenciales que se utilizaron en la implementación del editor:

Mostrar propiedades del componente: Visualiza las propiedades del componente deseado para su futura modificación en caso requerido.

Modificar propiedades de estilos CSS: Permite al usuario modificar las propiedades CSS de cada componente de la ribbon, mostrando al mismo tiempo el código CSS correspondiente a las propiedades modificadas y el diseño del componente.

Guardar Estilo: Guarda el estilo en formato ".css" en la dirección que el usuario especifique.

Abrir Estilo: Permite al usuario seleccionar un estilo previamente creado y continuar modificando las propiedades CSS de los componentes.

Crear Nuevo Estilo: Crear un nuevo estilo en la misma área de trabajo.

Cerrar estilo: Brinda la opción de cerrar el estilo, mostrando un mensaje con la opción de cerrar sin guardar, guardar o simplemente cancelar la acción.

2.3 Diseño de la interfaz

La interfaz tiene una facilidad de uso, pues posee una ribbon con 3 pestañas donde aparecen once componentes a los que se les va a modificar sus propiedades de estilos CSS. El panel de propiedades en la parte superior derecha muestra las propiedades correspondientes al componente que esté seleccionado. En la siguiente figura se refleja lo antes mencionado, en este caso ha sido seleccionado el botón *QPushButton*:

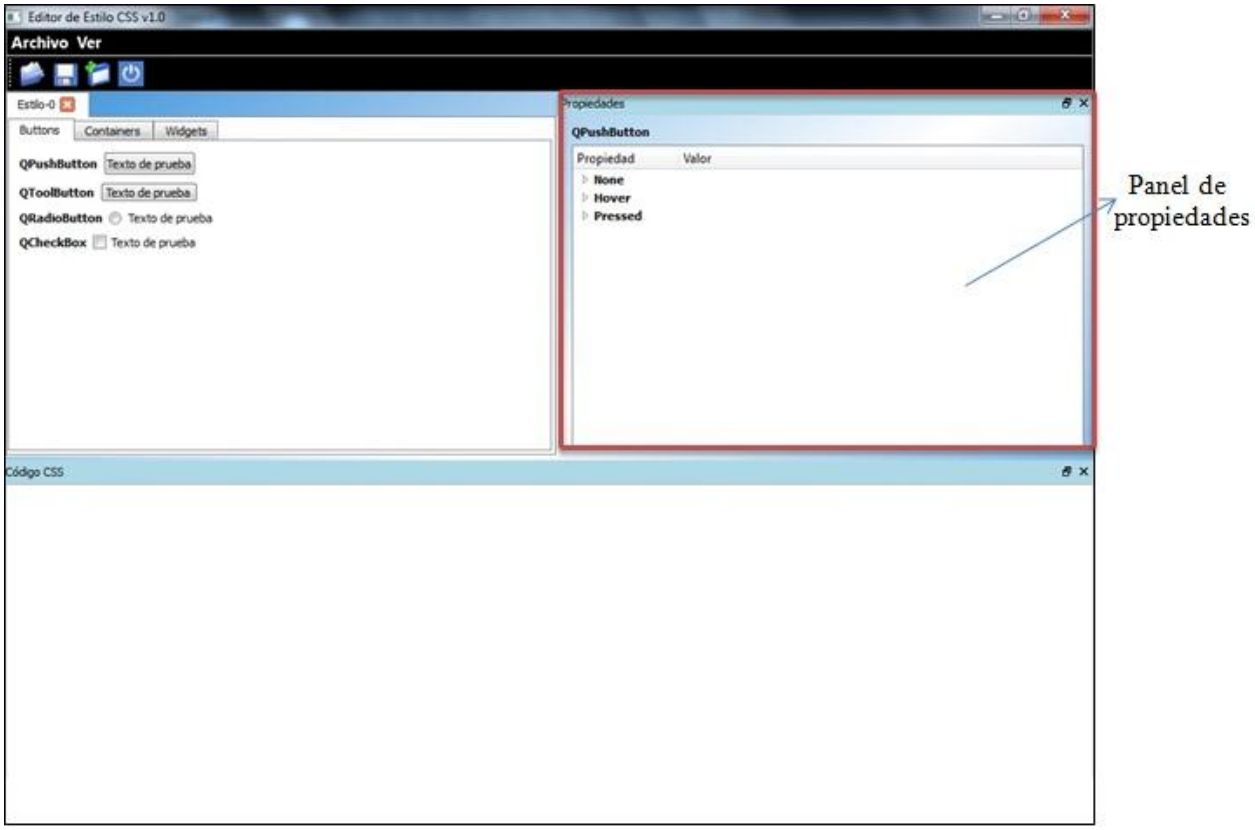


Fig. 8 Muestra del panel de propiedades correspondiente al botón QPushButton.

En la parte inferior de la aplicación aparece el área donde se va a mostrar el código generado al mismo tiempo que se va modificando las propiedades del componente seleccionado. En la siguiente figura se muestra el código generado, correspondiente a las propiedades de estilos que le fueron modificadas al componente QPushButton.

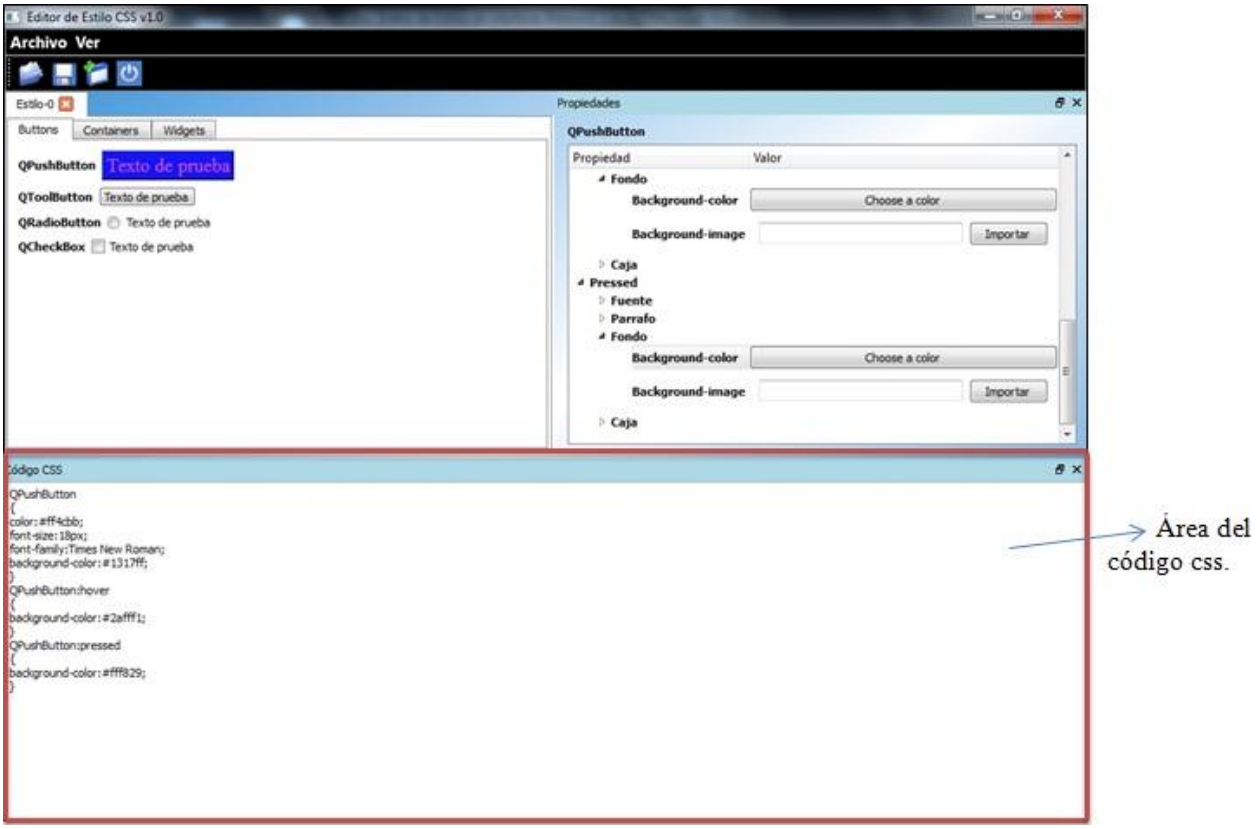


Fig. 9 Muestra del área donde se genera el código CSS.

El editor brinda la posibilidad de guardar el código en formato ".css" para luego ser cargado en otra aplicación, además de la opción de abrir un nuevo estilo o un estilo que ya este generado para continuar modificando sus propiedades.

La barra de menú proporciona dos menús desplegables. Es normalmente visible en todo momento y siempre es accesible desde el teclado, así que todos los comandos disponibles en la aplicación, están disponibles de igual forma en la barra de menú.

2.4 Framework de desarrollo

Esta biblioteca tiene en gran medida una amplia comunidad de desarrollo que le brinda soporte y posee muchas funcionalidades de alto rendimiento, además de contar con una amplia documentación.

Al instalar el paquete de desarrollo de software de Qt, el código que se escriba puede ser compilado sin cambio alguno en el entorno de trabajo deseado, sin importar si es Windows o GNU/Linux; sólo se debe copiar el código y compilarlo con Qt Creator, el cual se encuentra disponible para ambas plataformas. Se puede mencionar también que existen numerosos foros en Internet donde brindan ayuda en línea y en los que los programadores exponen dudas o problemas que son resueltos la mayoría de las veces en muy corto tiempo.

2.5 Modelo de dominio

A continuación se muestra el modelo de dominio que servirá como punto de partida para los posibles usuarios, ofreciendo mayor comprensión de todos los términos utilizados en la elaboración del editor de estilos para interfaces ribbon y que se utilizará como guía para el futuro diseño del sistema.

Desarrollador: Usuario que interactúa con el sistema.

QtCreator: IDE para el lenguaje C++, integra las herramientas QtAssistant y QtDesigner.

GUI Ribbon: Estilo de propiedades CSS a componentes de la ribbon.

QtRibbon: Editor de estilos CSS para interfaces ribbon propuesto.

Componentes: Componentes del *framework* Qt.

Propiedades de estilo: Propiedades de estilo CSS para modificar los componentes de la ribbon.

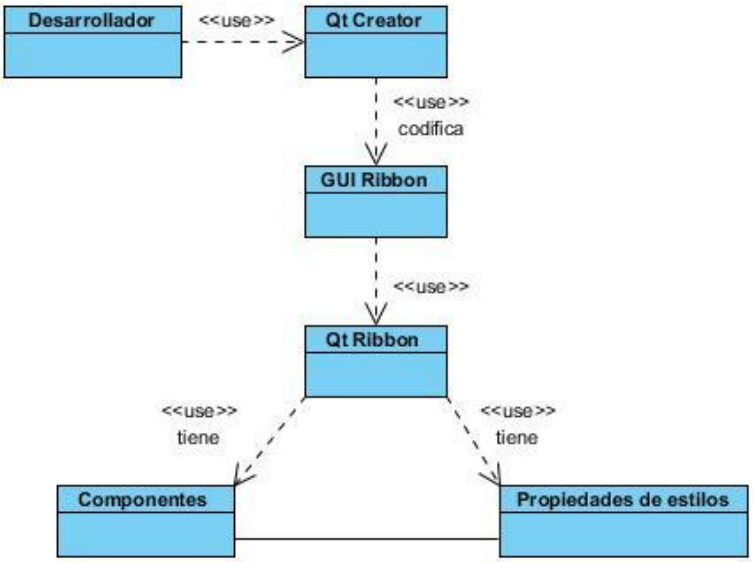


Fig. 10 Modelo de Dominio.

2.6 Requisitos del software

Los requisitos constituyen capacidades o condiciones que el sistema debe cumplir [11]. Estos facilitan el entendimiento entre los usuarios y los desarrolladores del sistema. A continuación se exponen los requisitos funcionales por los que se regirá el sistema y los no funcionales que exponen las características de la aplicación.

2.6.1 Requisitos Funcionales

Los siguientes requisitos establecen las funcionalidades e instrucciones que el sistema debe cumplir en su implementación [11].

El sistema debe permitir:

RF1. Seleccionar componente visual.

RF2. Modificar propiedades de estilos CSS.

RF3. Exportar código CSS generado.

RF4. Abrir estilo.

RF5. Crear nuevo estilo.

2.6.2 Requisitos No Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener [12]. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. A continuación se describen los requerimientos que presenta el editor de estilos para interfaces ribbon:

Requerimiento de Usabilidad: Los usuarios deben tener conocimientos básicos de programación.

Requerimiento de Software: Se recomienda el sistema operativo Windows 7, Ubuntu 12.04 o superior.

Requerimiento de Hardware: Debe tener una memoria RAM mínima de 512 MB.

Requerimiento de Apariencia o Interfaz Gráfica de Usuario: La interfaz gráfica de usuario debe proporcionar de forma coherente y sencilla, interactividad para todas las funcionalidades de la aplicación.

Requerimiento de Diseño e Implementación: Se utiliza el lenguaje de implementación C++ y el *framework* multiplataforma Qt para desarrollar la interfaz gráfica.

2.7 Modelo de Caso de Uso del Sistema

En esta sección se definen los actores del sistema, los casos de uso del sistema y la representación mediante un diagrama de casos de uso utilizando las facilidades que brinda el UML.

2.7.1 Actores del Sistema

Los actores representan entidades que interactúan con el sistema. Un actor no es parte del sistema, es un rol de un usuario, que puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema, y se beneficia con los resultados de las funcionalidades del mismo [13]. En el caso del editor de estilos para interfaces ribbon interactúa solo un actor que se puntualiza a continuación:

Tabla 3 Actor del Sistema

Actores	Justificación
Desarrollador	Es el que se beneficiará con las funcionalidades que brinda el editor de estilos para interfaces ribbon: modificar las propiedades de estilos CSS de los componentes de la ribbon que se brinda generando al mismo tiempo un código en formato ".css" que luego será exportado para su futura utilización en otras aplicaciones.

2.7.2 Diagrama de Casos de Uso del Sistema

A continuación se muestra el diagrama de casos de uso correspondiente al sistema. Este representa la relación entre el actor y los casos de uso definidos a partir de los requerimientos funcionales.

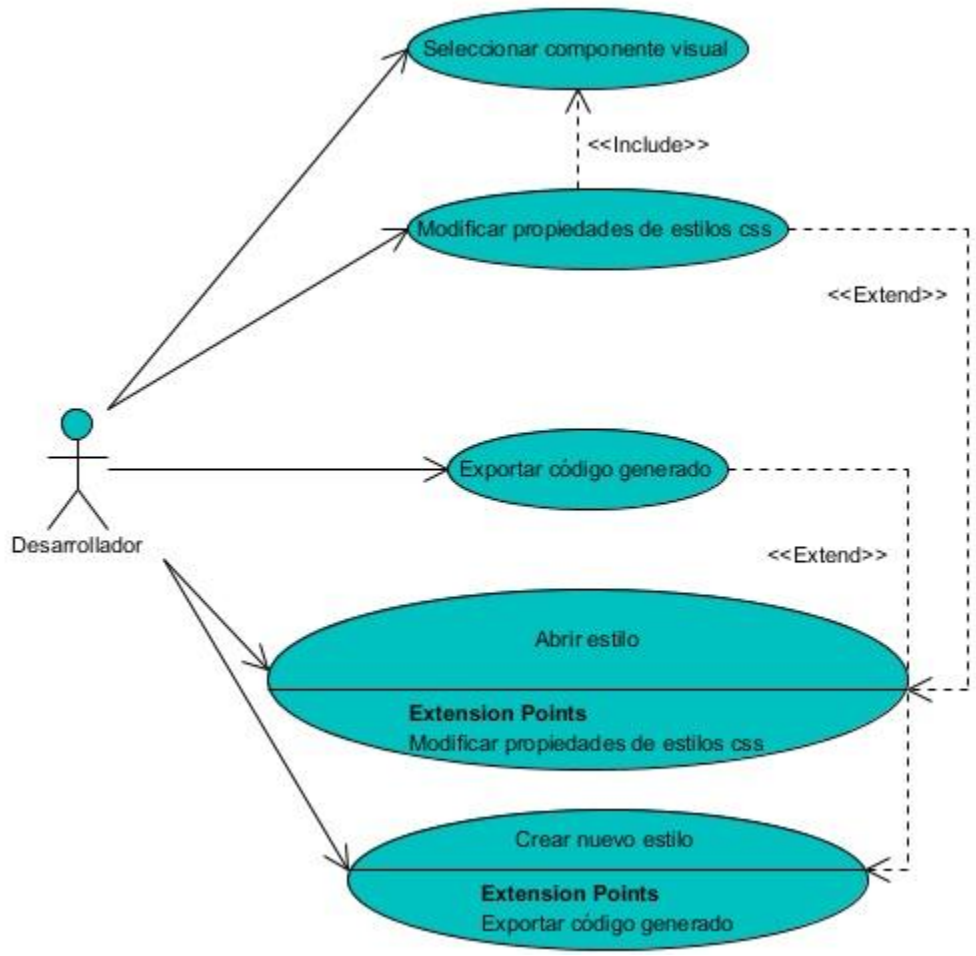


Fig. 11 Diagrama de Casos de Uso del Sistema.

2.7.3 Descripción de los Casos de Uso del Sistema

Resulta muy importante una descripción textual minuciosa de cada caso de uso, pues se logra entender con mayor facilidad la funcionalidad asociada a cada caso de uso, no es suficiente una representación gráfica mediante un diagrama de casos de uso. Es necesaria una descripción detallada de cada uno de estos por separado. A continuación se muestra una serie de tablas que agrupan los flujos operacionales de cada funcionalidad definida anteriormente.

Tabla 4 Descripción CU Seleccionar Componente Visual

Caso de Uso		
Nombre	Seleccionar componente visual.	
Actores	Desarrollador.	
Propósito	Mostrar las propiedades de estilos CSS del componente que ha sido seleccionado.	
Resumen	Se inicia cuando el desarrollador selecciona el componente al cual desea modificar sus propiedades de la ribbon que se muestra en el área de trabajo.	
Referencias	RF1.	
Precondiciones	Esté abierto al menos un estilo.	
Curso Normal de los Eventos		
Acción del Actor	**Respuesta del sistema**	
1. Selecciona el componente del cual quiere ver sus propiedades.	1.1 Muestra las propiedades en el Panel de Propiedades del componente que ha sido seleccionado.	
Poscondiciones	Se visualizan las propiedades correspondientes al componente seleccionado.	
Prioridad	Crítica.	

Tabla 5 Descripción CU Modificar Propiedades de Estilos CSS

Caso de Uso	
Nombre	Modificar propiedades de estilos CSS.
Actores	Desarrollador.
Propósito	Modificar las propiedades de estilos CSS asociadas al componente seleccionado.
Resumen	Se inicia cuando el desarrollador selecciona el componente al cual desea modificar sus propiedades de estilo CSS y este realiza los cambios que desee en el panel de propiedades.
Referencias	RF2.
Precondiciones	Debe haberse realizado el CU Seleccionar componente visual.
Flujo de Eventos	
Flujo Normal "Modificar propiedades de estilos CSS"	
Acción del Actor	**Respuesta del sistema**
1. Selecciona la pestaña de la ribbon que incluya el componente que desea seleccionar.	1.1 Muestra las propiedades del componente seleccionado, en el Panel de Propiedades.
2. Modifica las propiedades de estilo CSS	2.1 El sistema guarda los cambios realizados.

hasta lograr el diseño requerido.	2.2 El sistema genera un código al mismo tiempo que los valores se van modificando.
Poscondiciones	Se genera el código CSS de las propiedades del componente correctamente y al mismo tiempo que son modificadas.
Prioridad	Crítica.
Interfaz	

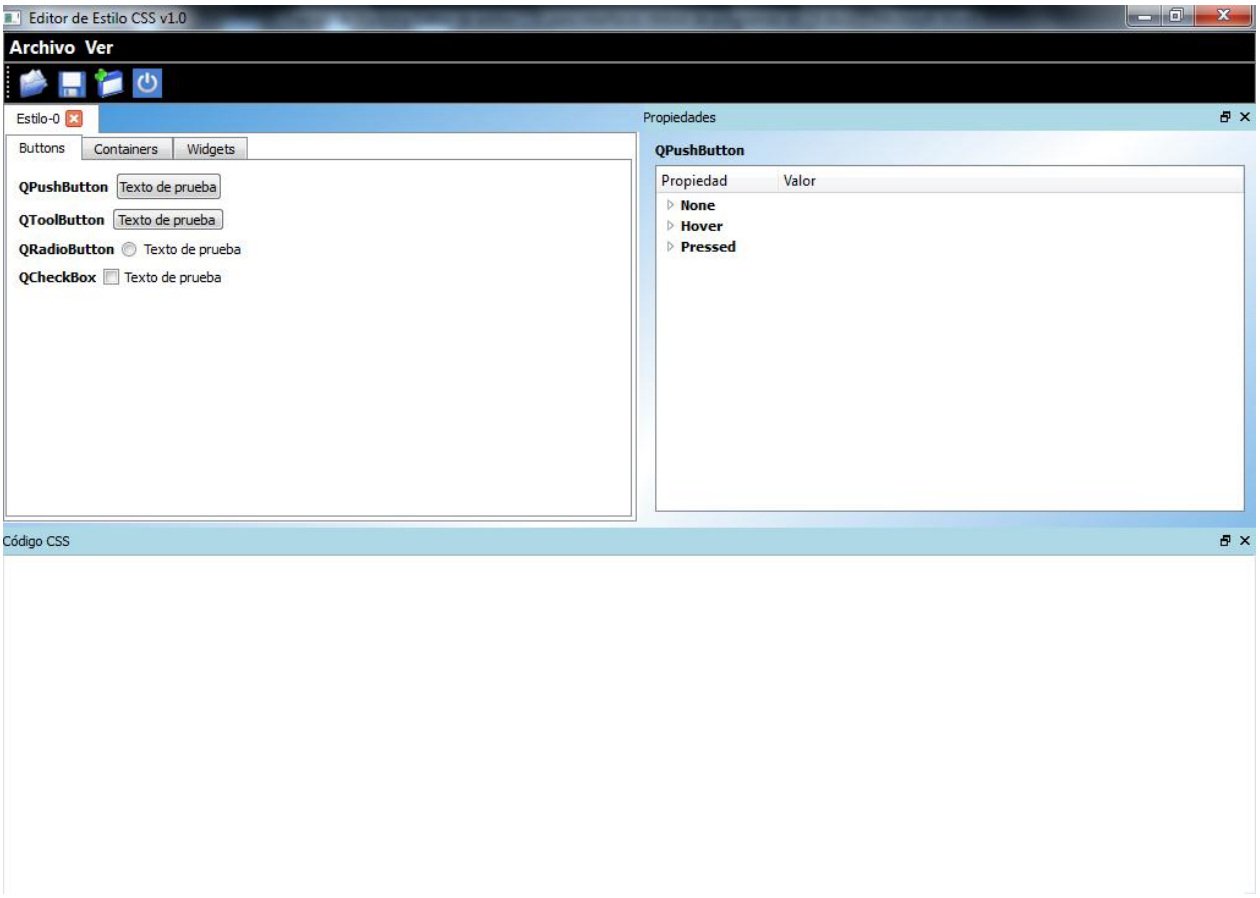
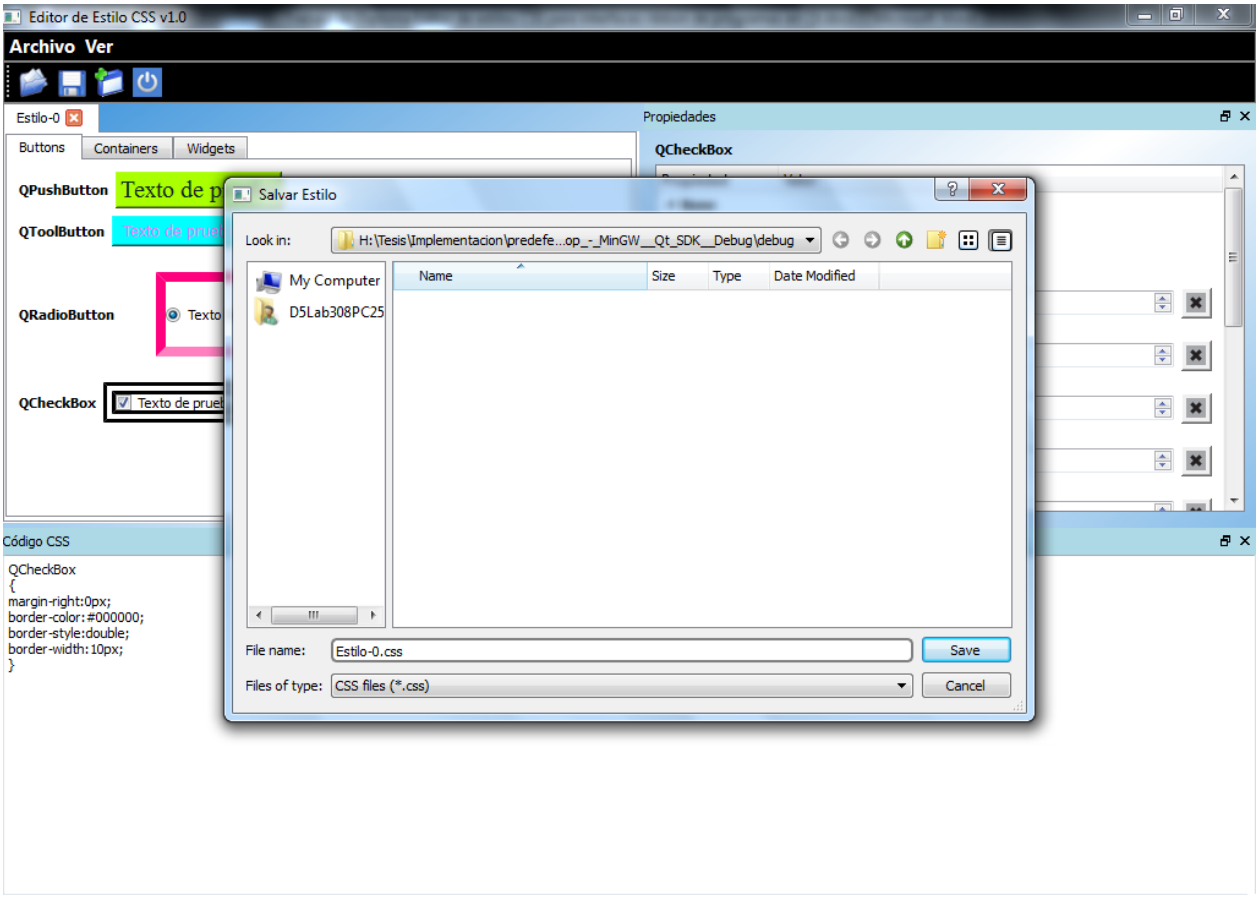


Tabla 6 Descripción del CU Exportar código generado

Caso de Uso	
Nombre	Exportar código generado.
Actores	Desarrollador.
Propósito	Exportar el código CSS para poder ser utilizado en otras aplicaciones.
Resumen	Se inicia cuando el desarrollador selecciona la opción “Guardar”, introduce el nombre del fichero y la dirección donde desea guardarlo.
Referencias	RF3.
Precondiciones	Debe haberse realizado el CU Modificar propiedades de estilos CSS.
Curso Normal de los Eventos	
Acción del Actor	Respuesta del sistema
1. Selecciona la opción “Guardar”.	1.1 Muestra una ventana para introducir la dirección y nombre con que se quiere guardar el fichero.
2. Introduce el nombre y la dirección donde desea que éste se encuentre.	2.1 El sistema guarda los valores entrados en la dirección especificada.
Poscondiciones	El código CSS se guarde en la dirección dada.
Prioridad	Crítica.

Interfaz


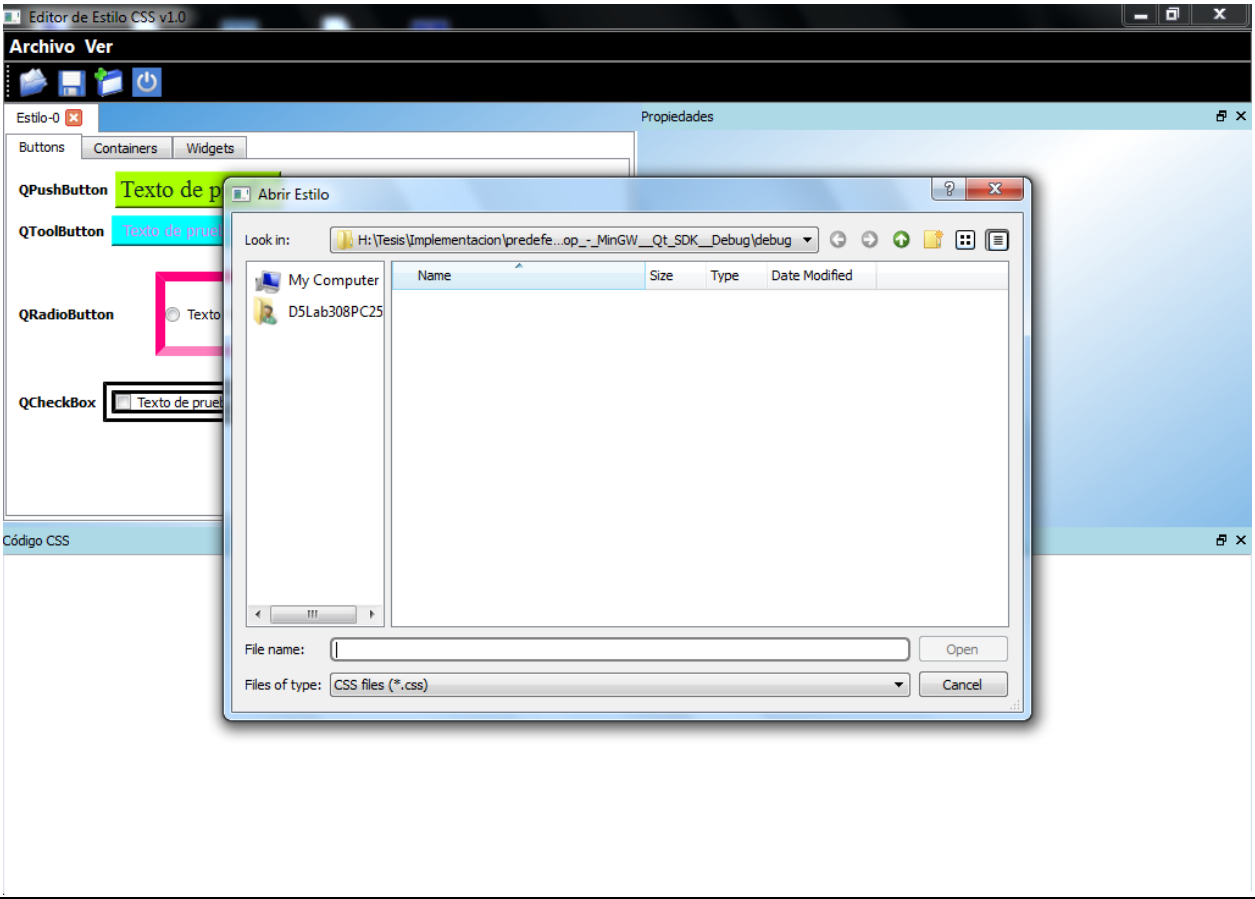
Flujo Alterno “Exportar código generado”	
Actor	**Sistema**
1. Si el usuario selecciona la opción “Cancelar”	1.1 El sistema cancela la acción y cierra la ventana de guardar estilo.

Tabla 7 Descripción del CU Abrir estilo

Caso de Uso	
Nombre	Abrir estilo.
Actores	Desarrollador.
Propósito	Cargar código en formato ".css" y si lo requiere continuar con la modificación del mismo.
Resumen	Se inicia cuando el desarrollador selecciona la opción Abrir, mostrando una ventana para especificar el fichero en formato ".css" que se quiera cargar en la aplicación.
Referencias	RF4.
Curso Normal de los Eventos	
Acción del Actor	**Respuesta del sistema**
1. Selecciona la opción Abrir.	1.1 Muestra una ventana para seleccionar la dirección del fichero en formato ".css" que desea abrir.
2. Selecciona el fichero ".css" que desea abrir.	2.1 El sistema carga el nuevo estilo seleccionado.
Poscondiciones	Se cargue satisfactoriamente el estilo abierto.

Prioridad	Crítica.

Interfaz

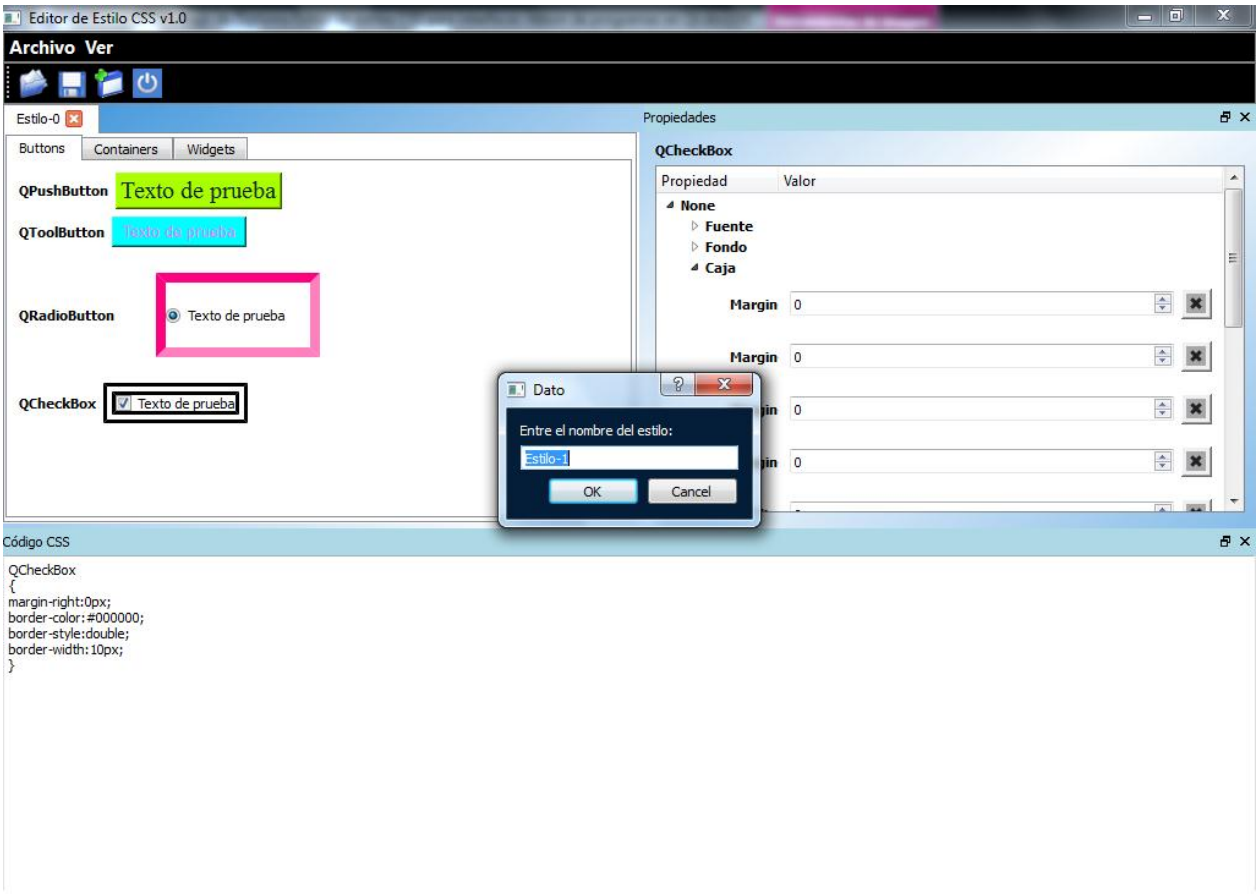


Flujo Alterno "Abrir estilo"

Actor	**Sistema**
1. Si el usuario selecciona la opción "Cancelar".	1.1 Se cancela la acción y se cierra la ventana de abrir un estilo.

Tabla 8 Descripción del CU Crear nuevo estilo

Caso de Uso		
Nombre	Crear nuevo estilo.	
Actores	Desarrollador.	
Propósito	Crear un nuevo estilo CSS en la misma área de trabajo.	
Resumen	Se inicia cuando el desarrollador selecciona la opción Nuevo para crear un nuevo estilo en una misma área de trabajo.	
Referencias	RF5.	
Curso Normal de los Eventos		
Acción del Actor		**Respuesta del sistema**
1. Selecciona la opción Nuevo.		1.1 Muestra un nuevo estilo.
Prioridad	Crítica.	
Interfaz		



Flujo Alterno “Nuevo estilo”	
Actor	**Sistema**
1. Si el usuario selecciona la opción “Cancelar”.	1.1 Se cancela la acción y se cierra la ventana de crear un nuevo estilo.

2.8 Diseño del Sistema

El diseño es un refinamiento del proceso de análisis del sistema enfocado en cómo el sistema cumple los objetivos propuestos y partiendo pues, de los requisitos no funcionales. El diagrama de clases del diseño se concibe a través de paquetes para prever una mayor claridad y comprensión de las clases que lo integran.

Los paquetes del diseño se utilizan para agrupar elementos del modelo del diseño relacionados con propósitos organizacionales y frecuentemente para administración de configuración, pueden ser relacionados a través de un diagrama de paquetes.

En la figura 14 se muestran los paquetes del diseño del sistema de forma general y las relaciones que existen entre ellos.

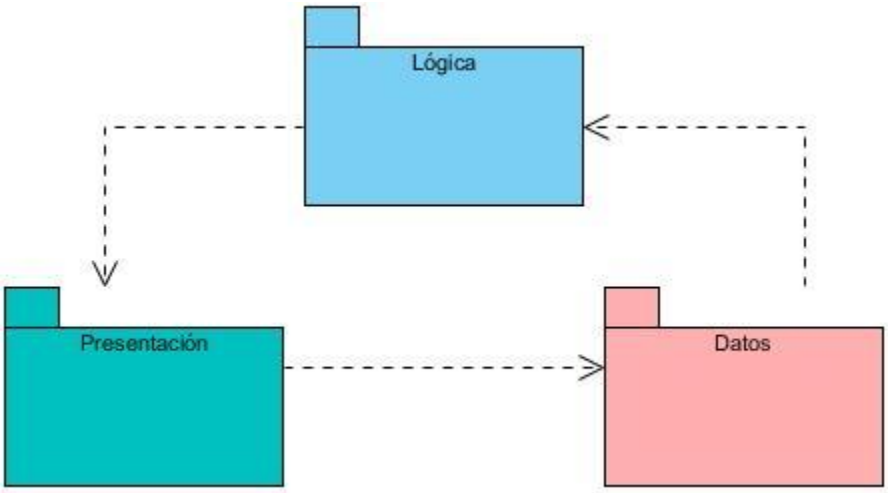


Fig. 12 Diagrama de Paquetes del Sistema.

2.8.1 Diagrama de Clases del Diseño

El diagrama de clases del diseño representa un modelo del objeto que describe la realización de los casos de uso, y sirve como una abstracción del modelo de implementación y su código fuente. De forma general la figura 15 se observa el diagrama de clases del diseño de la aplicación propuesta:

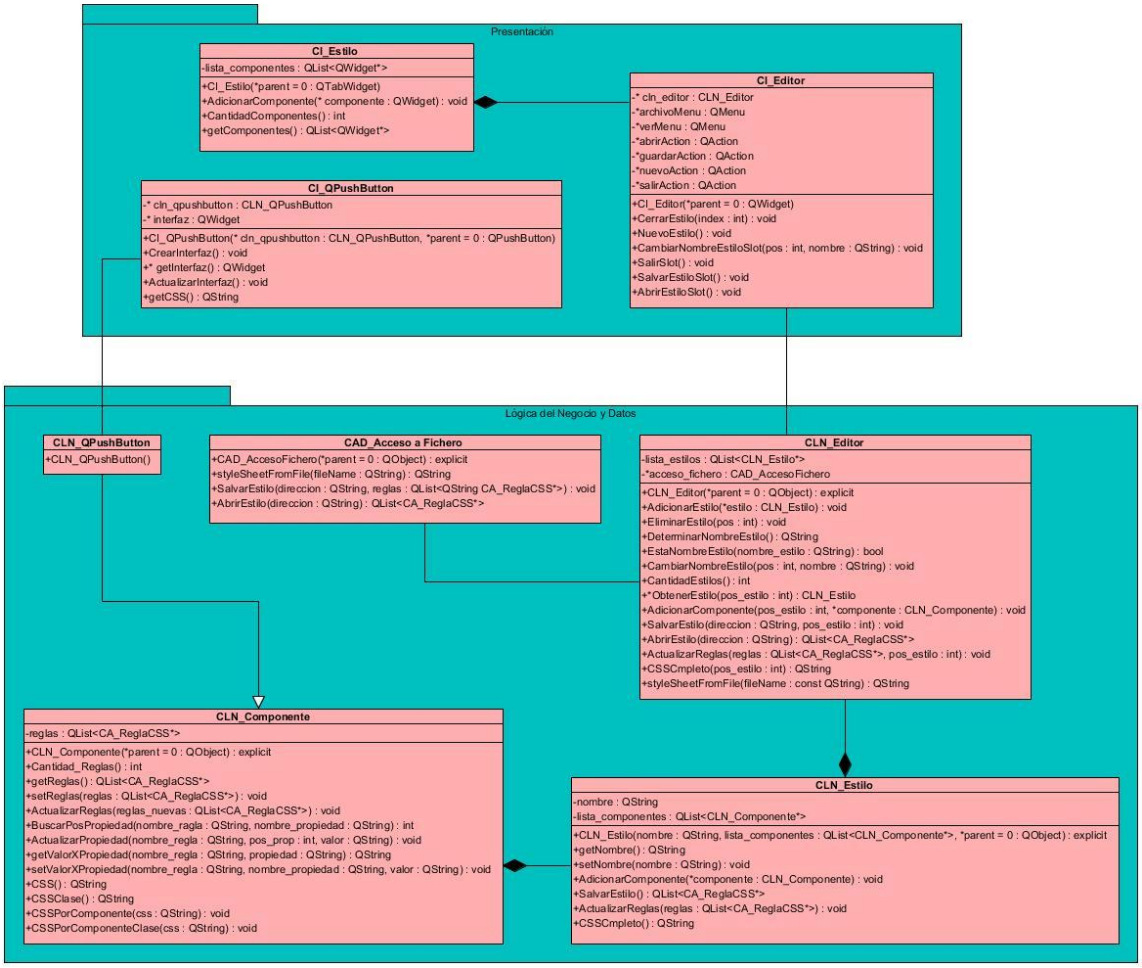


Fig. 13 Diagrama de Clases del Diseño.

2.8.2 Diagramas de Secuencia

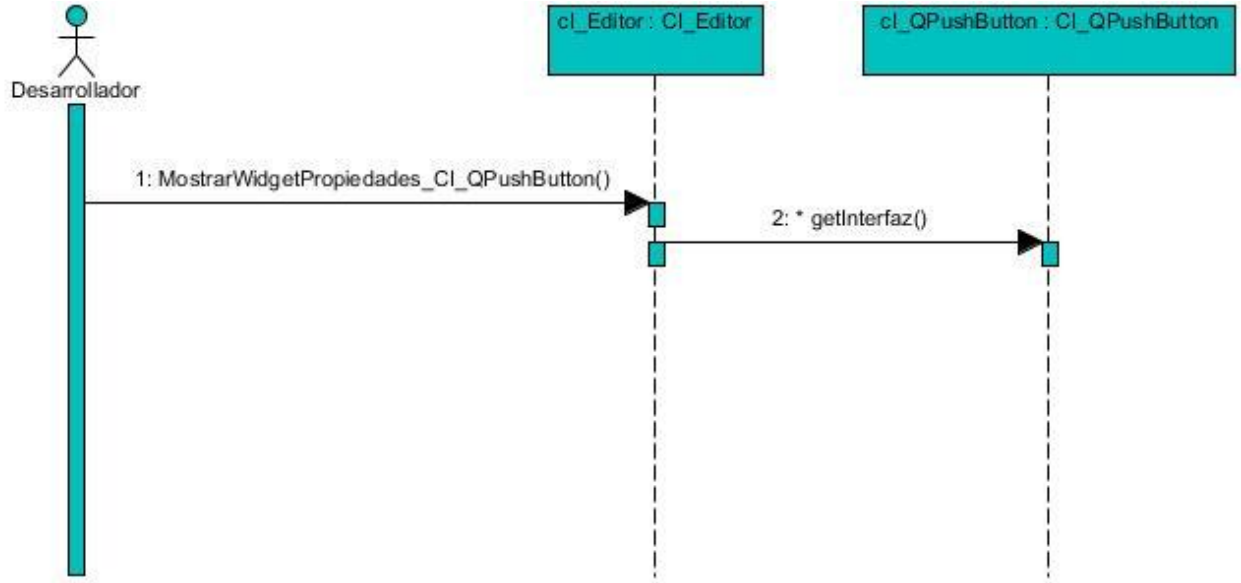


Fig. 14 Diagrama de Secuencia del Caso de Uso Seleccionar Componente.

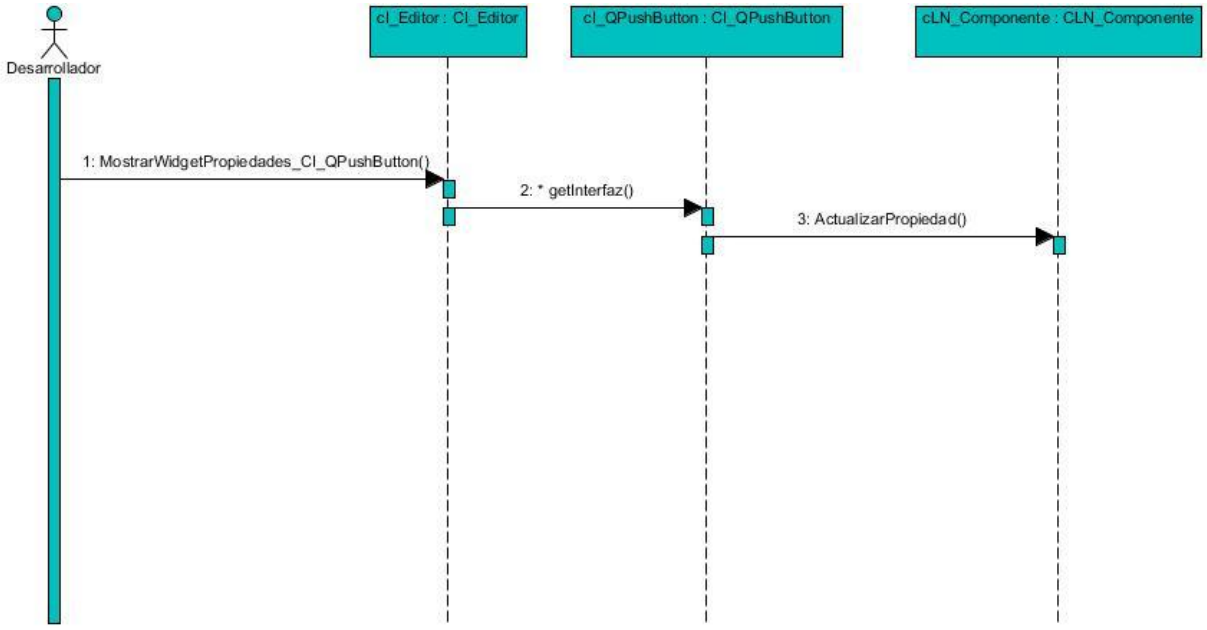


Fig. 15 Diagrama de Secuencia del Caso de Uso Modificar propiedades de los Componente.

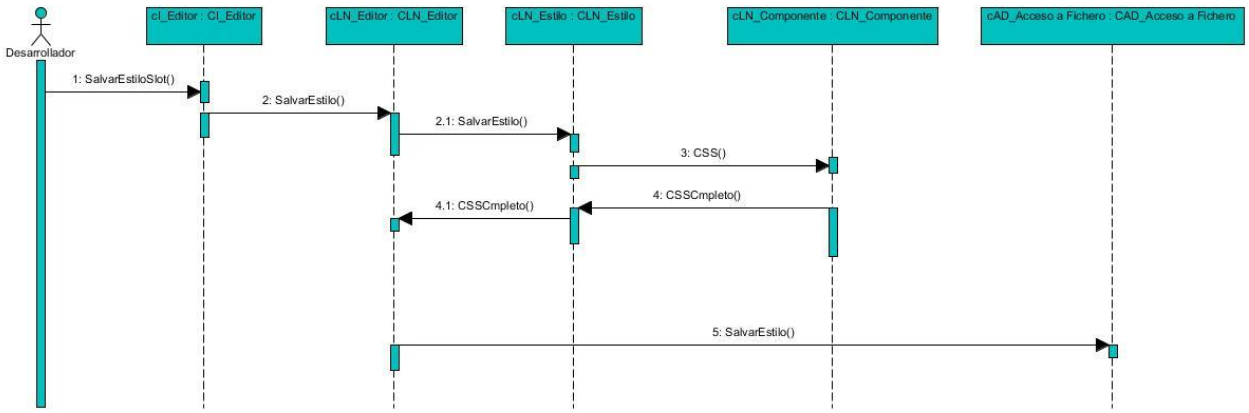


Fig. 16 Diagrama de Secuencia del Caso de Uso Exportar Código Generado.

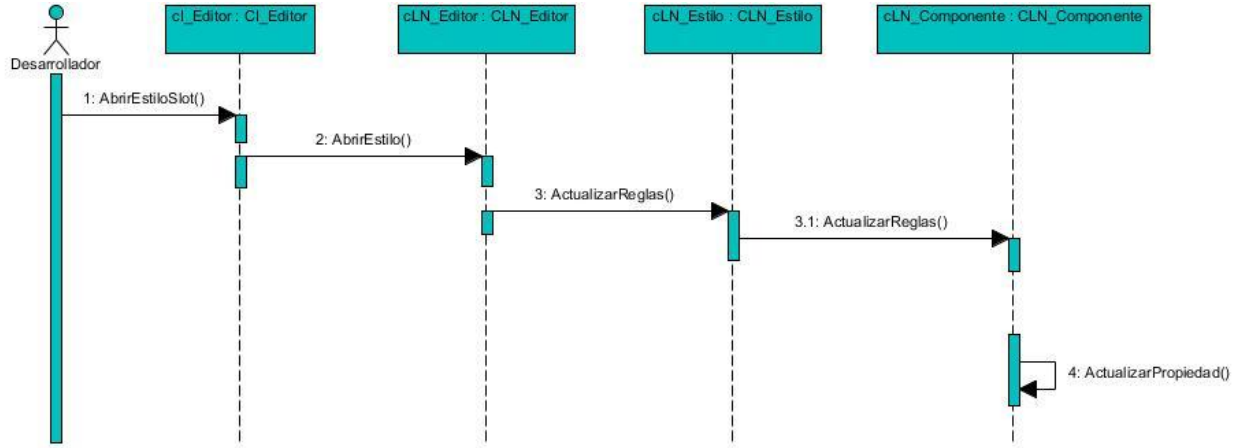


Fig. 17 Diagrama de Secuencia del Caso de Uso Abrir Estilo.

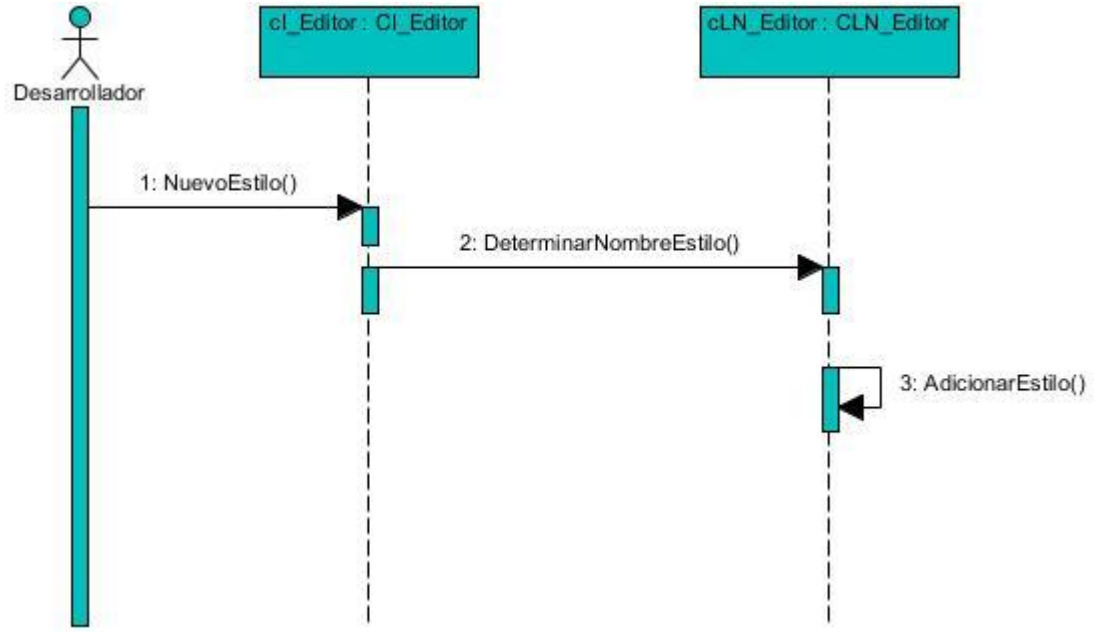


Fig. 18 Diagrama de Secuencia del Caso de Uso Nuevo Estilo.

2.9 Formato de fichero a exportar

Exportar un fichero es uno de los aspectos fundamentales que debe realizar el editor de estilos CSS. El editor proporciona al usuario la opción de modificar las propiedades de estilo CSS a uno de los componentes de la ribbon, generando al mismo tiempo un código CSS el cual va a ser exportado si el usuario desea.

El fichero a exportar es un fichero de texto plano que va a tener como extensión ".css". El texto que este va a incluir será el código CSS correspondiente a las propiedades de los componentes que fueron modificadas por el usuario, a continuación se muestra un pequeño ejemplo:

QLabel {

Font-size: 10pt; text-decoration: underline; color: black;

}

En la regla CSS mostrada anteriormente, se modifica el tamaño de fuente a 10 puntos, con texto subrayado y de color negro del label correspondiente para todas las instancias de la clase QLabel.

Luego de aclarada la sintaxis del mismo, solo resta guardar el fichero ".css" especificando la dirección que el usuario desee y el nombre con que se va a guardar el mismo.

A continuación se muestra un pequeño ejemplo usando el editor de estilos CSS.

QLabel

{ color:#ff577e;

font-size:20px;

font-family:New Century Schoolbook;

font-style:italic;

background-color:#0b0b0b; }

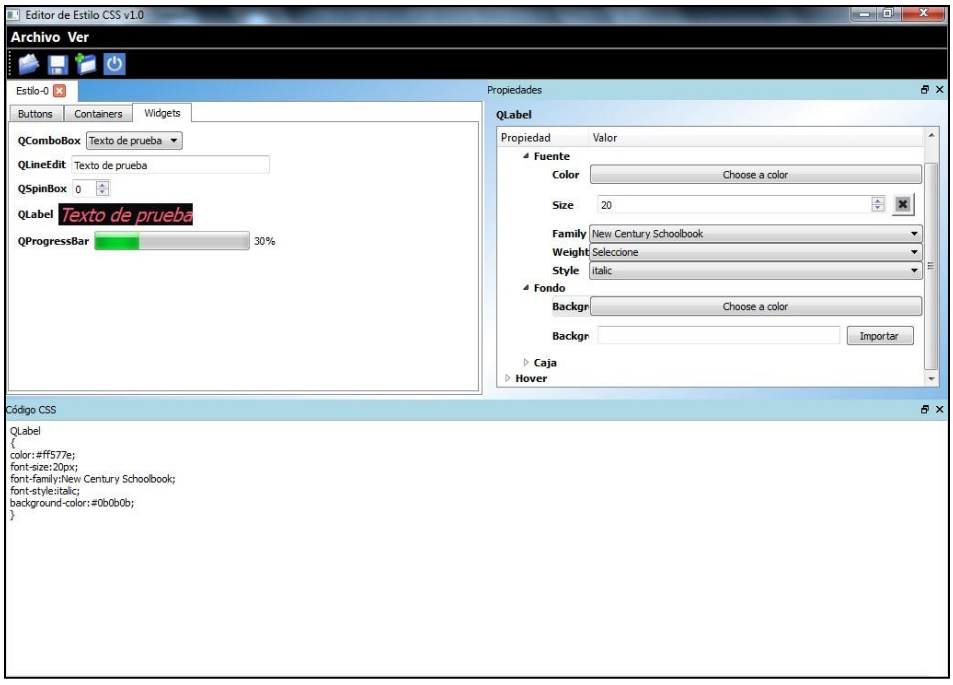


Fig. 19 Muestra de las propiedades CSS aplicadas al componente QLabel.

2.10 Herramienta, Lenguaje y Metodología

2.10.1 Metodología

Para guiar el proceso de desarrollo del sistema se utilizó el proceso unificado de desarrollo de software (RUP). Esta metodología además de ser un proceso de desarrollo de software bastante complejo, ofrece numerosas posibilidades de adaptación a sistemas de software sin tomar en cuenta el área a la que pertenece, organización, niveles de aptitud o dimensiones del proyecto. Otros aspectos que la hacen distintiva dentro del resto de las metodologías y la convierten en una más robusta es que: es **dirigida por casos de uso,** lo cual facilita que se siga el curso de los eventos a través de una serie de flujos de trabajos que tienen como inicio los casos de uso, que van dirigiendo todo el desarrollo del sistema con su maduración, **centrada en la arquitectura,** pues describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente, **iterativo e incremental,** lo que supone que cada proyecto o producto pueda dividirse y subdividirse en partes y que cada una de ellas se desarrolle de manera iterativa encerrando actividades de todos los flujos de trabajo, evolucionando de forma incremental a medida que se van uniendo los resultados para obtener el producto final, todo de forma planificada.

2.10.2 Herramientas de desarrollo

Como herramienta de modelado se empleó Visual Paradigm, para el diseño, cubriendo el desarrollo de software desde el paso de los requerimientos a través de los modelos de diseño, pruebas y mantenimiento.

Para la creación de la interfaz gráfica de usuario se empleó el *framework* Qt, lo que posibilita la portabilidad de la aplicación hacia diferentes sistemas operativos, facilitando en gran medida el desarrollo de nuevos componentes gráficos. Sus características son:

- ✓ Con el mismo código base, permite desplegar el sistema en múltiples plataformas.
- ✓ Producir aplicaciones de alto rendimiento con apariencia nativa.
- ✓ La concentración de los desarrolladores en la producción de código y no en las particularidades del sistema operativo.
- ✓ Acceso total al código fuente para revisión y modificación.

2.10.3 Lenguaje de modelado

UML es un lenguaje de modelado visual que se usa para especificar, construir, documentar y visualizar artefactos de un sistema de software. El mismo está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Su objetivo es visualizar, especificar, construir y documentar los artefactos que se crean durante el proceso de desarrollo.

Este lenguaje de modelado está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. UML incluye conceptos semánticos, notación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.

2.10.4 Lenguaje de programación

Como lenguaje de programación se utilizó C++, lenguaje por excelencia para las aplicaciones de realidad virtual que hace uso eficiente del paradigma de Programación Orientada a Objetos. Permite un excelente control de la memoria y buena administración de los recursos de la computadora. Dentro de las principales ventajas que presenta el lenguaje C++ se encuentran:

Difusión: al ser uno de los lenguajes más empleados en la actualidad, posee gran número de usuarios y tiene una excelente bibliografía.

Versatilidad: C++ es un lenguaje de propósito general, se puede emplear para resolver cualquier tipo de problema.

Portabilidad: se encuentra estandarizado, por tanto, el mismo código fuente puede ser compilado en diferentes plataformas.

Eficiencias: C++ es uno de los lenguajes más rápidos en tiempo de ejecución.

Herramientas: existe gran cantidad de compiladores, depuradores y bibliotecas de clases basadas en este lenguaje.

CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DEL SISTEMA.

El presente capítulo cubre el flujo de implementación realizado para la primera iteración del sistema propuesto. El resultado principal de la implementación, es la obtención de un fichero ”.css” que incluya un nuevo estilo de la ribbon. Este flujo especifica cómo van a estar desplegadas físicamente las distintas partes del sistema y mediante qué protocolos se comunicarán. Posteriormente se toman casos de prueba para validar los resultados alcanzados, según la realización de los pasos que define la metodología RUP para el desarrollo de software en la fase de pruebas.

3.1 Implementación

El resultado principal de la implementación, es la obtención de los componentes, dentro de los que se incluyen ficheros, ejecutables, una tabla y las dependencias existentes entre estos. Además, este flujo especifica cómo van a estar ubicadas físicamente las distintas partes del sistema.

3.1.1 Diagrama de componentes

Un componente no es más que el empaquetamiento físico de los elementos del modelo de diseño, cada componente define una interfaz que describe su funcionalidad y forma de empleo. El diagrama de componentes muestra la organización y las dependencias entre un conjunto interconectado de estas unidades, además de cubrir la vista estática ejecutable de la implementación de la aplicación, permite conocer a los desarrolladores y clientes la estructura física que tiene el sistema y cómo se relacionan sus partes. A continuación se muestra el diagrama correspondiente del sistema propuesto:

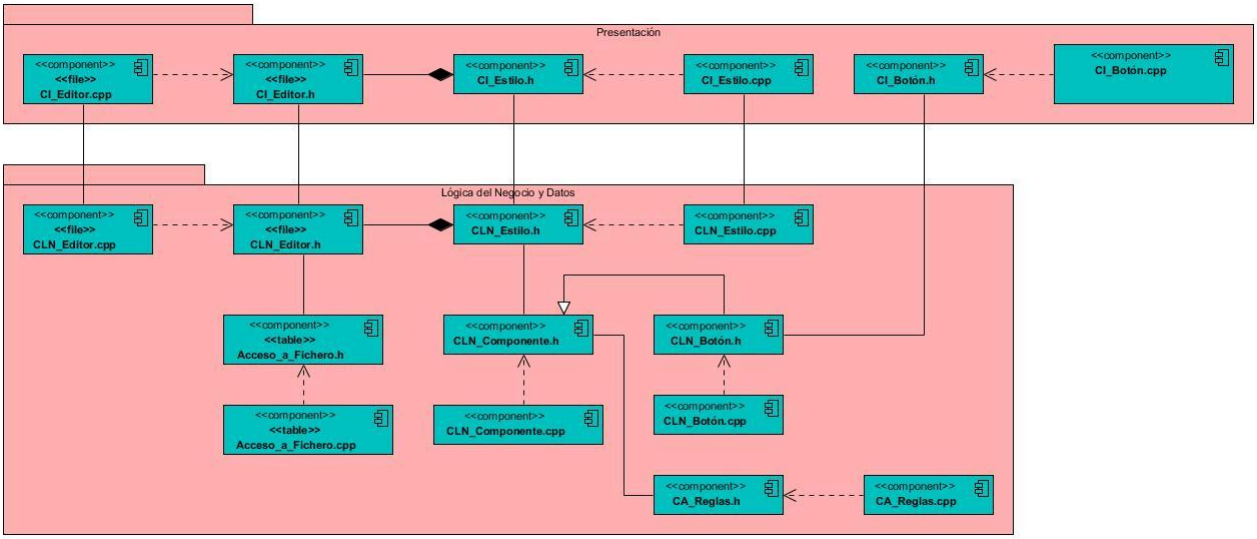


Fig. 20 Diagrama de Componentes.

3.2 Pruebas de la solución

Una vez generado el código fuente es necesario realizar las pruebas del sistema para detectar y corregir la mayor cantidad de errores en busca de elevar la calidad del software. Para lograr dicho objetivo se utilizan técnicas de prueba que permitan diseñar casos de prueba con una alta probabilidad de encontrar errores. Estas técnicas facilitan una guía sistemática para diseñar pruebas que se encargan de comprobar la lógica interna, la interfaz, el comportamiento y el rendimiento del software [14].

La técnica seleccionada para realizarle las pruebas al editor para la creación de un fichero CSS de forma segura es a través de las pruebas de caja negra**.** Las pruebas de caja negra, también

denominadas como pruebas de comportamiento, se centran en los requisitos funcionales del software. Son pruebas que se llevan a cabo sobre la interfaz del software, obviando la lógica interna y la estructura del programa. A partir de las pruebas de caja negra se pretenden encontrar los siguientes tipos de errores:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores en la interfaz.
- ✓ Errores en estructuras de datos o en accesos a bases de datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y de terminación.

3.2.1 Casos de prueba

Los casos de prueba especifican una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las cuales debe probarse. Los casos de prueba de caja negra pretenden demostrar que:

- ✓ Las funciones del software son operativas.
- ✓ La entrada se acepta de forma correcta.
- ✓ Se produce una salida correcta.
- ✓ La integridad de la información externa se mantiene.

Para la realización de los casos de prueba se utilizará específicamente el método de prueba partición equivalente. Este método se basa en la evaluación de clases de equivalencia para una condición de entrada, lo que permite representar un conjunto de estados válidos o inválidos [14]. En este caso la condición de entrada es lógica, lo que define una clase de equivalencia válida y una no válida. A continuación se muestran las pruebas realizadas a los casos de uso.

DCP[7]-Caso de uso seleccionar componente visual

Descripción general

[7] DPC: Diagrama de Casos de Prueba.

El caso de prueba se inicia cuando el usuario selecciona un componente visual incluido dentro de la ribbon.

Condiciones de ejecución

Debe estar al menos un estilo abierto para visualizar los componentes de la ribbon y proceder a la acción de seleccionar uno de ellos.

DCP-Caso de uso seleccionar componente visual

Tabla 9 Caso de uso seleccionar componente visual

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**
EC[8] 1.1 Seleccionar componente.	El sistema muestra una ribbon prediseñada que contiene tres pestañas, éstas incluyen once componentes. El usuario selecciona el componente presionando clic izquierdo del mouse sobre el mismo.	Visualiza las propiedades de estilo CSS correspondientes al componente seleccionado.	Se presiona clic en la pestaña que contiene el componente que se desea seleccionar. Se presiona clic en el componente.

[8] EC: Escenario.

DCP-Caso de uso modificar propiedades de estilos CSS.

Descripción general

El caso de prueba se inicia a partir de que el usuario realice algún cambio en las propiedades correspondientes al componente seleccionado.

Condiciones de ejecución

Debe haberse seleccionado un componente de la ribbon.

Tabla 10 Caso de uso modificar propiedades de estilos CSS

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**
EC 1.2 Modificar propiedades de estilos CSS.	El sistema muestra en el panel de propiedades las correspondientes al componente que ha sido seleccionado. El usuario procede a cambiar los valores de acuerdo al diseño que quiera lograr.	Al realizarse cambios en las propiedades del componente se muestran en el diseño del mismo automáticamente.	Se presiona clic en la pestaña que contiene el componente que se desea modificar. Se selecciona el componente. Se modifican los valores de las propiedades correspondientes al componente.

DCP-Caso de uso Exportar código generado

Descripción general

El caso de uso inicia a partir de que se presiona la opción Guardar sobre el menú de Archivos o el ícono correspondiente en la barra de herramientas, se introduce el nombre del fichero a exportar y la dirección donde se desea guardar.

Tabla 11 Caso de uso exportar código generado

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**
EC 1.3 Exportar código generado.	El sistema brinda la opción de guardar el estilo realizado.	Al seleccionar la opción Guardar se muestra una ventana para entrar el nombre y la dirección donde se desea guardar el fichero ".css" generado.	Se presiona clic izquierdo en la opción Guardar del menú de Archivos o sobre el ícono Guardar de la barra de herramientas. Se introduce el nombre del fichero a exportar y se selecciona la dirección donde se desee guardar.

DCP-Caso de uso abrir estilo

Descripción general

El caso de uso inicia a partir de que se presiona la opción Abrir sobre el menú de Archivos o el ícono correspondiente en la barra de herramientas, se selecciona el fichero que se desea cargar.

Tabla 12 Caso de uso abrir estilo

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**
EC 1.4 Abrir estilo.	El sistema brinda la opción de abrir un estilo en formato ”.css” para ser modificado si así lo requiere.	Muestra una ventana para seleccionar el fichero ”.css” que se desea cargar en la aplicación. Luego de seleccionar el fichero se carga el estilo en la ribbon.	Se presiona clic izquierdo sobre la opción Guardar del menú de Archivos o sobre el ícono de la barra de herramientas. Se selecciona el fichero ”.css” que se desea cargar y luego la opción Abrir.

DCP-Caso de uso nuevo estilo

Descripción general

El caso de uso inicia a partir de que se presiona la opción Nuevo sobre el menú de Archivos o el ícono correspondiente en la barra de herramientas, se introduce el nombre del estilo nuevo.

Tabla 13 Caso de uso nuevo estilo

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo Central**
EC 1.4 Crear nuevo estilo.	El sistema brinda la opción de crear estilos nuevos.	Crea uno o varios estilos en una misma área de trabajo.	Se presiona clic izquierdo sobre la opción Nuevo del menú de Archivos o sobre el ícono de la barra de herramientas. Se introduce el nombre del nuevo estilo y se presiona la opción ¨ OK ¨

3.2.2 Resumen de las pruebas realizadas

Al finalizar la primera iteración de pruebas al sistema se encontraron 4 no conformidades, y se procedió a realizar una segunda iteración donde las mismas fueron corregidas arrojando resultados satisfactorios, debido a esto no se necesitó realizar una tercera iteración. En la Fig. 23 se representa un gráfico de barras para establecer una comparativa entre los resultados obtenidos luego de la primera y segunda iteración de pruebas, teniendo en cuenta cada caso de uso.

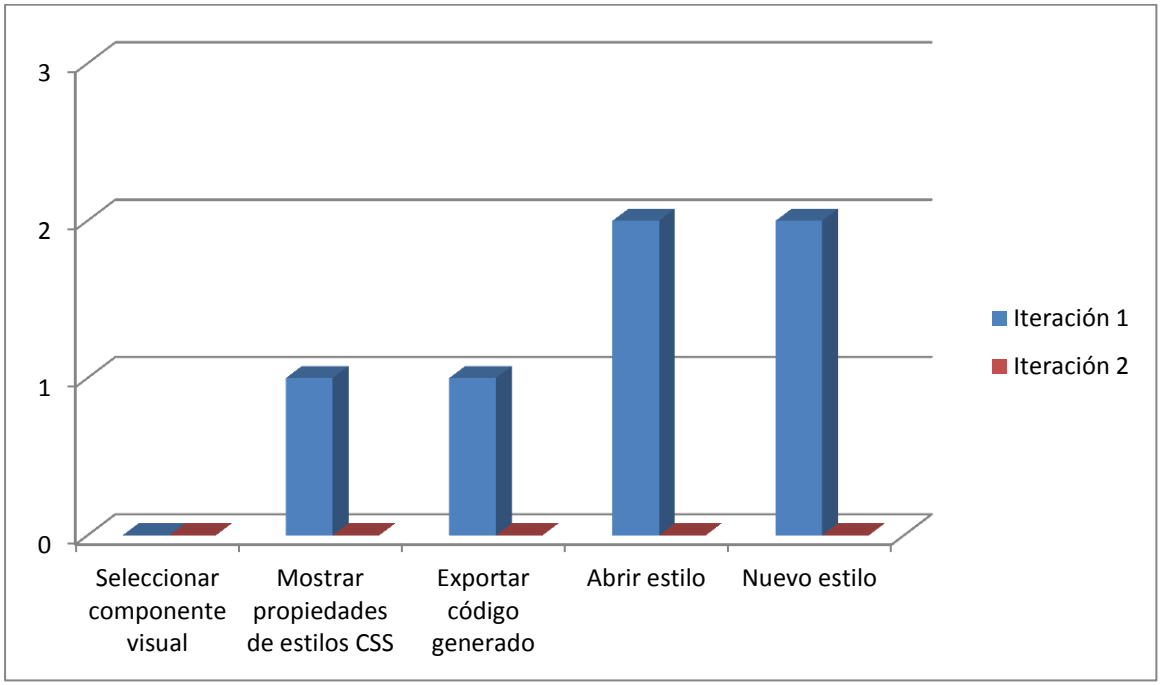


Fig. 21 Resultados de las Pruebas.

A partir de la realización de las pruebas, se validó que los requisitos funcionales definidos para el editor de estilos son totalmente operativos.

CONCLUSIONES

Como resultado del presente trabajo se obtuvieron las siguientes conclusiones:

- ✓ La posibilidad de visualizar en tiempo de ejecución los cambios realizados a las propiedades CSS de los componentes gráficos, sin necesidad de implementar a nivel de código fuente en C++ y CSS empleando el *framework* Qt, acelera el tiempo de diseño y prueba de los nuevos estilos.
- ✓ A partir de los resultados obtenidos de la aplicación de pruebas de caja negra, se comprobó que la solución cumple con todos los requisitos funcionales definidos.

RECOMENDACIONES

Durante el desarrollo del trabajo ha surgido una idea que podría implementarse en versiones posteriores del sistema, de forma que se logren agregar o modificar las funcionalidades, para lo cual se recomienda:

- ✓ Añadir otros componentes del *framework* Qt a la ribbon prediseñada en el editor de estilos CSS.

BIBLIOGRAFÍA

1. Cuaderno Digital De Juan Sebastian Romero. *Cuaderno Digital De Juan Sebastian Romero.* [En línea] 31 de Enero de 2012. [Citado el: 24 de Mayo de 2013.] http://www.Conceptos de Ribbon/Cuaderno Digital De Juan Sebastian Romero INTERFAZ RIBBON Y INTERFAZ METRO.htm.

2. Opus Planet Conceptos Básicos. *Opus Planet Conceptos Básicos.* [En línea] [Citado el: 24 de Mayo de 2013.] http:\\www.Conceptos de Ribbon/OPUS PLANET - Opus Planet Conceptos Básicos.htm.

3. Windows. [En línea] [Citado el: 10 de Diciembre de 2012.] http://msdn.microsoft.com/en-us/library/windows/desktop/cc872782.aspx.

4. free(code). [En línea] [Citado el: 11 de Diciembre de 2012.] http://windows.freecode.com/users/qtitan..

5. Gutiérrez, David González. *Tutorial de Qt4 Designer y QDevelop.* 2008.

6. Qt Project. [En línea] [Citado el: 11 de Diciembre de 2012.] http://qt-project.org/wiki/Category:Tools::QtCreator_Spanish#7dab7df03f6770f5ec001cbda5343d57..

7. Assistant del Framework Qt.

8. elWebmaster. [En línea] [Citado el: 14 de Diciembre de 2012.] http://www.ucware.com/juststyle.

9. elWebmaster. [En línea] [Citado el: 14 de Diciembre de 2012.] http://www.hostm.com/css/.

10. Ray Rischpater, Daniel Zucker. *Beginning Nokia Apps Development. Qt and HTML5 for Symbian and MeeGo.* 2010.

11. Ecured. *Ecured.* [En línea] [Citado el: 9 de Enero de 2013.] http://www.ecured.cu/index.php/Requisitos_de_Software.

12. eumed.net. [En línea] [Citado el: 1 de Marzo de 2013.] http://www.eumed.net/libros-gratis/2010b/698/Requisitos%20no%20funcionales.htm.

13. Fractal group. [En línea] [Citado el: 1 de Marzo de 2013.] http://fractalgroup.co/recursos/index.php?option=com_content&view=article&id=26:actores-casos-de-uso&catid=2:recursos&Itemid=17.

14. Pressman. Ingeniería de Software. [aut. libro] Roger Pressman. *Un enfoque práctico. Quinta edición. Madrid : s.n., 2002.* 2002.

15. Jasmin Blanchette, Mark Summerfield. *C++ GUI Programming.* 2006.

16. Molkentin, Daniel. *Qt4 The art of Building Qt Applications.* 2006.

17. Thelin, Johan. *Foundations of Qt Development.* 2006.

18. Summerfield, Mark. *Advanced_QT_Programming.* 2011.

19. Alan Ezust, Paul Ezust. *An Introduction to Design Patterns in C++ with Qt 4.* 2011.

20. Anónimo. *Aprenda_Qt4_Hoy_Mismo.* 2010.

GLOSARIO DE TÉRMINOS

C

CDSEM: Centro de Diseño y Simulación de Estructuras Mecánicas.

C++: Lenguaje de programación, la intención de su creación fue extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos.

CSS: Hojas de estilos en cascada. Es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla. CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas web al mismo tiempo.

G

GALBA-CAD: Software creado por el equipo CDSEM y miembros de la línea Visualización Científica.

H

HTML: Lenguaje de marcado hipertextual. Hace referencia al lenguaje de marcado predominante para la elaboración de páginas web que se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>).

I

ICVT: Industria China Venezolana de Taladros.

O

OpenGL: Es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D.

Q

Qt: Tecnología que proporciona un juego de herramientas y elementos gráficos para la creación de interfaces y aplicaciones multiplataforma.

QML: Es un lenguaje basado en JavaScript creado para diseñar aplicaciones enfocadas a la interfaz de usuario. Se usa principalmente para aplicaciones móviles, donde la entrada táctil, las animaciones fluidas y una buena experiencia de usuario son cruciales.

QtQuick: Kit de Interfaz de usuario creado por Nokia junto al *framework* QT.

R

Ribbon: Cinta de opciones.

S

SDK: Kit de Desarrollo de Software. Un conjunto de herramientas de desarrollo que le permite a un programador crear aplicaciones para un sistema concreto, por ejemplo ciertos paquetes de software, *frameworks*, plataformas de hardware, computadoras, videoconsolas, sistemas operativos, entre otros.

W

Widgets: Elementos de control.

X

XML: Lenguaje de marcas extensible. Permite definir la gramática de lenguajes específicos para estructurar documentos grandes. Da soporte a bases de datos, siendo útil cuando varias aplicaciones se deben comunicar entre sí o integrar información.