



Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Mención Ingeniería de Software

HP Views Designer Assistant

Autor:

Alberto Gutiérrez Pérez

Tutores:

Margarita Gonzalo Tasis

Miguel Ángel Benítez Andrades

Agradecimientos

Quiero agradecer de una forma especial a mi familia y amigos, por apoyarme de forma incondicional y por no dudar de mí en ningún momento, desde el comienzo de la carrera hasta la finalización de este proyecto. A Margarita Gonzalo Tasis, mi tutora en la Universidad, por la atención, la paciencia y la ayuda que me ha proporcionado durante todo el proyecto. A HP en general, por brindarme la oportunidad de realizar este proyecto de forma conjunta, y concretamente a Miguel Ángel Benítez Andrades, por el apoyo proporcionado y por concederme una gran libertad a la hora de realizar este proyecto.

Muchas Gracias.

Resumen

Actualmente para crear una interfaz en formato QML es necesario el uso de programas de diseño, haciendo que se convierta en una tarea repetitiva. Además, no existe la posibilidad de realizar pruebas y solo se puede conocer el resultado una vez se finalice el proceso.

Para agilizar el proceso, este proyecto intenta proporcionar una forma rápida de crear y editar determinados tipos de interfaces. Al tratarse de un proyecto desarrollado conjuntamente con HP, el tipo de interfaces que se van a generar son las que se instalan en los dispositivos de impresión. Tras seguir una serie de pasos, el sistema genera una interfaz en formato QML, así como una vista previa del resultado obtenido. De esta forma, se permite generar de forma masiva interfaces de uso genérico con un alto grado de personalización.

Abstract

Currently to create an interface in QML format is necessary to use design programs, turning it into a repetitive task. Furthermore, there is no possibility of testing, and only the result can be known once the process is completed.

To speed up the process, this project attempts to provide a quick way to create and edit certain types of interfaces. As a project developed in jointly with HP, the type of interfaces to be generated are those that are installed on their printing devices. After following a series of steps, the system generates an interface in QML format, as well as a preview of the result obtained. This way, it is possible to generate massively generic interfaces with a high degree of customization.

Índice General

Capítulo 1. Introducción.....	15
Capítulo 2. Contexto.....	19
Qt	21
Proyecto Qt.....	23
Ejemplos de uso.....	23
QML	23
Modelo QML.....	24
Creación interfaces QML	25
Qt Creator	25
Código Fuente.....	26
Capítulo 3. Estudio Previo	27
Conclusiones del estudio. Tecnología elegida.	30
Capítulo 4. Desarrollo del proyecto.....	31
4.1. Proceso de desarrollo	33
4.2. Gestión de riesgos	34
4.3. Integrantes y responsabilidades.....	38
4.2. Planificación Temporal.....	38
4.3. Vista en detalle	40
4.4. Vista gráfica	41
4.5. Seguimiento del proyecto.....	42
Capítulo 5. Análisis.....	43
5.1. Requisitos Funcionales	45
5.2. Requisitos no funcionales	46
5.3. Actores.....	46
5.4. Casos de Uso	46
5.5. Modelo de dominio.....	51
Capítulo 6. Diseño.....	53
6.2. Arquitectura General.....	55
6.3. Diagramas de secuencia.....	58
Creación interfaz tipo Menú.....	62
Creación interfaz tipo Información	65
Creación interfaz tipo Tinta.....	68
Edición	70

Capítulo 7. Implementación	71
Navegación básica	74
Creación de ventanas de tipo Menú	76
Creación de ventanas de tipo Información.....	77
Creación de ventanas de tipo Nivel de Tinta	77
Corrección de errores I.....	78
Corrección de errores II.....	79
Finalización	79
Capítulo 8. Pruebas	81
8.1. Pruebas de caja negra.....	83
Capítulo 9. Tecnología Empleada	89
9.1. C++	91
9.4. Qt Creator.....	91
Capítulo 10. Manuales	93
10.1. Manual de instalación	95
10.2. Manual de usuario	97
10.3. Ventana principal.....	98
Opciones principales.....	98
Opciones Secundarias.....	99
10.4. Creación.....	99
10.4.1. Menú	99
Parámetros generales del menú	100
Parámetros de cada icono	102
10.4.2. Información.....	104
Parámetros generales de la pantalla de información.....	104
Parámetros de cada tarjeta.....	107
10.4.3. Tinta	109
10.5. Edición	111
10.6. Barra de “Menú”	113
10.6.1. Home.....	113
10.6.2. Ajustes.....	113
10.6.3. Ayuda	114
Capítulo 11. Conclusiones	117
11.1. Líneas futuras.....	119

Bibliografía	121
Anexos	127
Anexo I – Informes de progreso	129
Hito 1. 10/02/2017 – 24/02/2017	129
Hito 2. 25/02/2017 – 17/03/2017	133
Hito 3. 18/03/2017 – 29/03/2017	134
Hito 4. 30/03/2017 – 15/04/2017	135
Hito 5. 17/04/2017 – 26/04/2017	136
Hito 6. 27/04/2017 – 12/05/2017	137
Apéndices	139
Contenido del CD-ROM	141

Índice de figuras

Figura 1 Logo Qt.....	21
Figura 2 Ejemplo fichero QML.....	24
Figura 3 Modelo QtQuick.....	25
Figura 4. IDE Qt Creator.....	26
Figura 5 Esfuerzo en actividades según la fase. RUP.....	33
Figura 6 Estimación del producto <<probabilidad x impacto>> para evaluar riesgos.....	34
Figura 7. Planificación tareas del proyecto en detalle.....	40
Figura 8 Planificación tareas del proyecto de forma gráfica.	41
Figura 9. Diagrama de casos de uso.....	46
Figura 10. CU 1 – Crear Interfaz.....	47
Figura 11. CU 2 – Editar Interfaz.....	48
Figura 12. Diagrama de secuencia crear interfaz.....	49
Figura 13. Diagrama de secuencia editar interfaz.....	50
Figura 14. Diagrama modelo de dominio.	52
Figura 15. Arquitectura General.....	56
Figura 16. Modelo en diseño.....	57
Figura 17. Diagrama de secuencia inicio.....	58
Figura 18. Diagrama de secuencia Ir a Ajustes.....	59
Figura 19. Diagrama de secuencia ayuda.....	59
Figura 20. Diagrama de secuencia Ir a inicio.....	60
Figura 21. Diagrama de secuencia Ajustes.....	61
Figura 22. Diagrama de secuencia Creación General Menu.....	62
Figura 23. Diagrama de secuencia Propiedades Generales Menú.....	63
Figura 24 Diagrama de secuencia Creación Icono Menú.....	64
Figura 25 Diagrama de secuencia Creación General Información.....	65
Figura 26. Diagrama de secuencia Propiedades Generales Información.....	66
Figura 27. Diagrama de secuencia Propiedades Tarjeta Información.....	67
Figura 28. Diagrama de secuencia Creación General Tinta.....	68
Figura 29. Diagrama de secuencia Seleccionar Propiedades Tinta.....	69
Figura 30. Diagrama de secuencia Editar.....	70
Figura 31 Organización del proyecto por paquetes.....	73
Figura 32 Ejemplo del esquema QMainWindow Qt.....	74
Figura 33 Icono lenguaje c++.....	91
Figura 34 Icono Qt Creator.....	91
Figura 35 Mensaje bienvenida instalador.....	95
Figura 36 Selector directorio instalador.....	95
Figura 37 Selección menú inicio instalador.....	96
Figura 38 Progreso instalador.....	96
Figura 39 Animación inicio.....	97
Figura 40 Ventana Antes de Comenzar.....	97
Figura 41 Ventana principal de la aplicación.....	98
Figura 42 Icono creación menú.....	100
Figura 43 Propiedades generales del menú.....	100
Figura 44 Botones selección de fondo Menú.....	100
Figura 45 Mensajes de información menú.....	101
Figura 46 Zona edición número de iconos del menú.....	101

Figura 47 Botones edición número de iconos del menú.....	101
Figura 48 Opción de título en iconos.....	101
Figura 49 Selector disposición iconos del menú.....	102
Figura 50 Mensaje de error del menú.....	102
Figura 51 Selector posición de los iconos.....	102
Figura 52 Botón seleccionar icono menú.....	103
Figura 53 Mensaje de error imagen del icono.....	103
Figura 54 Campo título del icono.....	103
Figura 55 Posición del icono.....	103
Figura 56 Icono creación interfaz información.....	104
Figura 57 Propiedades generales interfaz información.....	104
Figura 58 Botones selección de fondo interfaz información.....	105
Figura 59 Mensajes información interfaz información.....	105
Figura 60 Edición número de elementos interfaz información.....	105
Figura 61 Botones edición número de elementos información.....	105
Figura 62 Selector disposición tarjetas.....	106
Figura 63 Mensaje de error información.....	106
Figura 64 Selector iconos información.....	106
Figura 65 Selector disposición propiedades tarjeta.....	106
Figura 66 Título tarjeta.....	107
Figura 67. Botón seleccionar imagen.....	107
Figura 68 Icono añadido información.....	107
Figura 69 Mensaje de error tarjeta.....	108
Figura 70 Propiedades tarjeta.....	108
Figura 71 Añadir propiedad tarjeta.....	108
Figura 72 Editar propiedad tarjeta.....	108
Figura 73 Eliminar propiedad tarjeta.....	109
Figura 74 Icono creación interfaz tinta.....	109
Figura 75 Tamaño interfaz tinta.....	109
Figura 76 Edición número de elementos interfaz tinta.....	110
Figura 77 Botones edición número de elementos interfaz tinta.....	110
Figura 78 Selector disposición elementos tinta.....	110
Figura 79 Mensaje de error interfaz tinta.....	110
Figura 80 Selector colores niveles de tinta.....	111
Figura 81 Icono editar interfaz.....	111
Figura 82 Ventana seleccionar fichero a editar.....	112
Figura 83 Error de formato en el archivo a editar.....	112
Figura 84 Error en el fichero QML a editar.....	113
Figura 85 Confirmación volver a inicio.....	113
Figura 86 Directorio por defecto.....	113
Figura 87 Idioma de la aplicación.....	114
Figura 88 Restaurar valores.....	114
Figura 89 Mensaje de éxito ajustes.....	114
Figura 90 Ventana Acerca de.....	114
Figura 91 Icono de ayuda.....	115
Figura 92 Ventana principal de ayuda.....	115

Índice de tablas

<i>Tabla 1. Fuente: traducido de https://www.qt.io/licensing-comparison/.....</i>	<i>22</i>
<i>Tabla 2. Riesgo 001 – Problemas en la creación de ficheros QML.....</i>	<i>35</i>
<i>Tabla 3 Riesgo 002 – Modificación de los requisitos</i>	<i>35</i>
<i>Tabla 4 Riesgo 003 – Retraso en la planificación</i>	<i>36</i>
<i>Tabla 5 Riesgo 004 – Problemas con el diseño.....</i>	<i>36</i>
<i>Tabla 6. Riesgo 005 – Problemas con el desarrollo de software</i>	<i>37</i>
<i>Tabla 7 Riesgo 006 – Problemas con la estación de trabajo.....</i>	<i>37</i>
<i>Tabla 8 Integrantes y responsabilidades – Margarita Gonzalo Tasis.....</i>	<i>38</i>
<i>Tabla 9 Integrantes y responsabilidades – Miguel Ángel Andrades.....</i>	<i>38</i>
<i>Tabla 10 Integrantes y responsabilidades – Alberto Gutiérrez Pérez</i>	<i>38</i>
<i>Tabla 11. Planificación temporal.....</i>	<i>39</i>
<i>Tabla 12. RF – 001 Creación interfaz.....</i>	<i>45</i>
<i>Tabla 13. RF – 002 Modificación de interfaces.....</i>	<i>45</i>
<i>Tabla 14. RF – 003 Tipos de interfaz.....</i>	<i>45</i>
<i>Tabla 15. RF – 004 Introducción de parámetros</i>	<i>45</i>
<i>Tabla 16. RNF – 001 Formato QML</i>	<i>46</i>
<i>Tabla 17. RNF – 002 Estilo Asistente</i>	<i>46</i>
<i>Tabla 18 Actor 001 – Usuario Programador</i>	<i>46</i>
<i>Tabla 19. Prueba 001 - Home.....</i>	<i>83</i>
<i>Tabla 20. Prueba 002 - Ayuda.....</i>	<i>83</i>
<i>Tabla 21. Prueba 003 - Ajustes.....</i>	<i>83</i>
<i>Tabla 22. Prueba 004 – Restaurar Ajustes.....</i>	<i>83</i>
<i>Tabla 23. Prueba 005 – Acerca de.....</i>	<i>84</i>
<i>Tabla 24. Prueba 006 – Creación interfaz menú</i>	<i>84</i>
<i>Tabla 25. Prueba 007 – Cancelar creación menú</i>	<i>84</i>
<i>Tabla 26. Prueba 008 - Navegación menú.....</i>	<i>84</i>
<i>Tabla 27. Prueba 009 – Creación interfaz información.....</i>	<i>84</i>
<i>Tabla 28. Prueba 010 - Cancelar creación información</i>	<i>85</i>
<i>Tabla 29. Prueba 011 – Navegación información</i>	<i>85</i>
<i>Tabla 30. Prueba 012 – Creación interfaz tinta.....</i>	<i>85</i>
<i>Tabla 31. Prueba 013 – Cancelar creación tinta.....</i>	<i>85</i>
<i>Tabla 32. Prueba 014 – Navegación interfaz tinta</i>	<i>86</i>
<i>Tabla 33. Prueba 015 – Selección imagen de fondo</i>	<i>86</i>
<i>Tabla 34. Prueba 016 – Cancelar selección imagen de fondo</i>	<i>86</i>
<i>Tabla 35. Prueba 017 – Selección color de fondo</i>	<i>86</i>
<i>Tabla 36. Prueba 018 – Cancelar selección de color de fondo</i>	<i>87</i>
<i>Tabla 37. Prueba 019 – Edición interfaz</i>	<i>87</i>
<i>Tabla 38. Prueba 020 - Error edición</i>	<i>87</i>
<i>Tabla 39. Prueba 021 – Cancelar edición.....</i>	<i>87</i>

Capítulo 1. Introducción

El objetivo principal de este Trabajo de Fin de Grado se centra en la creación de un asistente guiado (“Wizard”) que nos ayude en la creación de una interfaz del tipo especificado por una serie de parámetros, en formato QML. Además, debe permitir la edición de las interfaces ya creadas con el propio asistente.

El motivo de la realización de este proyecto viene condicionado por la repetición de la tarea de crear interfaces en formato QML. Hasta ahora la creación de interfaces en este formato se realiza con programas de diseño generales. El uso de estos programas obliga al usuario a tener unos conocimientos previos sobre la tecnología. Esta herramienta facilita y acorta los procesos de creación y edición de interfaces QML. Además, elimina la necesidad de que el usuario conozca el lenguaje. En concreto se centra en tres tipos diferentes de interfaces, elegidos de forma específica dada la relación de este proyecto con la empresa HP. De este modo se pueden generar interfaces en formato QML de forma semiautomática sin la necesidad de usuarios experimentados.

En los siguientes capítulos se va a describir el proceso de implementación del proyecto. Se comienza con un breve apartado de información sobre las tecnologías QT y QML, los editores y herramientas existentes, así como software similar de creación de interfaces a de forma automatizada. Gracias a esta información se adquieren parte de las características e ideas con las que posteriormente se llevará a cabo el proyecto.

A continuación, se presentará el estudio previo realizado con el fin de decidir que tecnología es la más apropiada para la realización del proyecto. Este estudio se realiza dada la posibilidad de realizar el desarrollo del proyecto con diferentes tecnologías, así como diferentes lenguajes de programación. Al final del estudio se realiza un balance y se decide la tecnología a utilizar durante todo el proyecto.

Se aporta también una planificación detallada de las tareas, el tiempo estimado para una de ellas y una reducida información sobre si se ha completado en el tiempo establecido. En caso de que las tareas no se hayan completado en el tiempo previsto se añade la causa y la solución elegida con el fin de paliar el problema.

Tras esto, se muestra el resultado de las fases de análisis, diseño e implementación del sistema. Por último, se describen los manuales de usuario y de instalación, las conclusiones del trabajo realizado y la bibliografía consultada para el desarrollo del Trabajo de Fin de Grado.

Capítulo 2. Contexto

Actualmente existe una gran demanda de aplicaciones de usuario en diferentes sistemas. La utilización de diferentes sistemas operativos y plataformas sumado a la velocidad de cambio que caracteriza a la informática, hace que sea un reto conseguir desarrollar a tiempo y de forma estable una aplicación en múltiples plataformas.

Con el fin de resolver este problema se intentan desarrollar aplicaciones genéricas que no dependan del sistema en el que se encuentren. Ahí aparecen los frameworks multiplataforma que, a base de utilizar sistemas no nativos para el desarrollo de las aplicaciones, consiguen el objetivo deseado.

Uno de estos frameworks es Qt, el cual permite desarrollos para todos los sistemas operativos más utilizados como son: Windows, MacOs, Android o IOS entre otros.

Qt

Para poder entender el desarrollo del proyecto hay que hablar obligatoriamente de Qt. Todo el proyecto se ha llevado a cabo con las herramientas que este framework facilita. Es ampliamente utilizado en todo tipo de proyectos: software con interfaz gráfica, en sistemas informáticos embebidos para automoción, aeronavegación y aparatos domésticos como frigoríficos. [1][2]

Se trata de un framework multiplataforma que utiliza el C++ como lenguaje de programación nativo, aunque permite su utilización con otros lenguajes de programación a través de bindings o asociaciones. Entre estos lenguajes se encuentran algunos tan utilizados como:

- Python a través de PyQt/PySide [3]
- C# a través de Qyoto
- Java Qt Jambi



Figura 1 Logo Qt

Posee un API para gestionar las estructuras de datos tradicionales, acceso a base de datos SQL, gestión de hilos, soporte de red además del uso de manipulación de archivos y uso de diferentes formatos como XML. Qt se encuentra disponible para la mayoría de plataformas como, por ejemplo: sistemas tipo Unix, Apple Mac OS X, sistemas Microsoft Windows, Linux embebido, sistemas embebidos como PDA, Smartphone, etc. y para dispositivos que utilizan Windows CE10.

Aparece como biblioteca desarrollada por Trolltech en 1992 siguiendo un desarrollo basado en el código abierto, pero no completamente libre. Esta distinción se le otorga dado que, aunque permitía el desarrollo openSource utilizando una licencia de tipo Free Qt, no permitía redistribuir versiones modificadas de Qt. También permitía el desarrollar software cerrado siempre que previamente se hubiese adquirido una licencia comercial.

Posteriormente se utilizó para el desarrollo del escritorio KDE, lo que le proporcionó un gran reconocimiento y expansión otorgándole el título de posible candidato a convertirse en uno de los escritorios más populares de GNU/Linux. [4]

En junio de 2008 Nokia adquiere Trolltech y convierte el desarrollo en Qt en su principal plataforma de desarrollo de sus dispositivos. En enero de 2009, en la versión 4.5 de Qt, añaden otra opción de desarrollo, la LGPL haciendo que Qt sea aún más atractivo para para proyectos de código abierto no GPL y también para aplicaciones cerradas. [5]

Para poder desarrollar cualquier proyecto cuenta con un sistema triple de licencias:

- GPL v2/v3 para el desarrollo de software de código abierto
- LGPL licencia gratuita para aplicaciones comerciales
- QPL licencia de pago para aplicaciones comerciales

Para poder entender de una forma simplificada y clara las diferencias que proporciona cada una de las versiones se proporciona la siguiente tabla:

	LGPLV3	GPLV2/GPLV3	Licencia Comercial
Precio	Gratis	Gratis	Desde 79\$ al mes
Soporte Comunitario	✓	✓	✓
Servicio de asistencia oficial Qt			✓ ¹
Mantener la aplicación privada			✓
Posibilidad de mantener la aplicación privada con enlace dinámico	✓		✓
No obligación de proveer una copia de la licencia y reconocer explícitamente el uso de Qt			✓
No obligación de habilitar una copia del código fuente a los consumidores			✓
Aplicación de patentes de software	✓	✓	✓
Posibilidad de creación de dispositivos de consumo privada			✓
Funcionalidad general Qt Creator IDE	✓ ²	✓	✓
Herramientas de documentación	✓ ²	✓	✓
Herramientas de internalización	✓ ²	✓	✓
Qt Quick Designer	✓ ²	✓	✓

Tabla 1. Fuente: traducido de <https://www.qt.io/licensing-comparison/>

¹ Soporte no incluido en el nivel de licencia Start-up

² Herramienta autorizada bajo licencia GPLv3 aunque se permite su uso para desarrollo mediante LGPLV3 sin modificar la herramienta o, en caso de modificación, la necesidad de publicarla.

En marzo de 2011 Nokia comunica la venta de los servicios de Qt a la finlandesa Digia, que anuncia de forma inmediata su objetivo de realizar soporte para plataformas como Android, iOS, y Windows 8. En los años posteriores Qt va ganando importancia dentro de la empresa, hasta que en mayo de 2016 se convierte en una empresa totalmente independiente de Digia.

Proyecto Qt

Qt 5 se publica a finales de 2012 con uno de los mayores cambios dentro de la plataforma hasta la fecha, con QML y JavaScript jugando un rol principal y una gran mejora para aplicaciones gráficas. Además, el desarrollo de Qt 5 se amplía mediante el proyecto qt-proyect.org siguiendo los conceptos de *open governance*. [6][7]

Desde entonces se han publicado varias versiones y diferentes mejoras hasta la versión actual, Qt 5.9. [8]

Ejemplos de uso

Debido a las grandes ventajas que proporciona Qt son muchas las empresas que hacen uso de este framework, entre las que se encuentran algunas tan importantes como: Panasonic, Philips, Samsung, Siemens, Walt Disney Animation Studios, Electronic Arts, AMD, HP, etc.

Su utilización en aplicaciones de uso general se encuentra ampliamente extendido, es por ello por lo que algunas de las aplicaciones que se usan de forma masiva diariamente utilizan esta tecnología. Aplicaciones como: Google Earth, VirtualBox, VLC media player, Adobe Photoshop Album, Skype, algunas versiones de Spotify, etc. [7]

QML

Como se ha comentado anteriormente, el objetivo del presente proyecto se centra en desarrollar un asistente que ayude en la creación de interfaces para los dispositivos de impresión de HP. Al realizar el proyecto bajo la supervisión y con la colaboración de dicha empresa, el formato elegido para la realización de las interfaces es QML, lenguaje utilizado de forma común dentro de la empresa.

QML es una especificación de interfaz de usuario y lenguaje declarativo que permite definir interfaces de usuario en función de sus componentes y cómo interactúan estos entre ellos. Se trata de un lenguaje de alto nivel diseñado para permitir la interconexión de sus elementos de forma dinámica, además permite reutilizar y personalizar todos sus elementos. [9]

Ofrece una sintaxis similar a JSON, legible y declarativa, con soporte para expresiones JavaScript y asociaciones de propiedades entre elementos.

```
import QtQuick 2.0

Rectangle {
    id: page
    width: 500; height: 500

    Text {
        id: helloText
        text: "Hello world!"
        anchors.centerIn: parent
        font.pointSize: 24; font.bold: true
    }
}
```

Figura 2 Ejemplo fichero QML

Haciendo uso del módulo QtQuick, se pueden desarrollar interfaces de usuario en QML con todo tipo efectos animados y transiciones. Además, contamos con la posibilidad de conectar estas interfaces a porciones de código fuente desarrollado en C++ dada su compatibilidad con las librerías de este lenguaje. [10]

Un ejemplo de uso en aplicaciones de este tipo de interfaces se puede ver en la plataforma para dispositivos táctiles de Canonical: Ubuntu Phone, donde el lenguaje QML es uno de los pilares del sistema operativo.

Modelo QML

Una de las actividades básicas que realizan las aplicaciones basadas en la interacción con el usuario es la de formar y mostrar la información. Para realizar esta actividad QtQuick utiliza una variante del patrón Modelo-Vista-Controlador (MVC).

La parte correspondiente al controlador se desarrolla en código JavaScript o mediante código fuente, mayoritariamente en C++. La variación llega con el modelo y la vista dado que estos elementos se encuentran separados en 3 elementos diferentes en las interfaces de tipo QML. Estos 3 elementos son modelo, vista y delegado. Debemos de intentar no confundir el delegado con el controlador del patrón MVC dado que no realizan la misma función. [11]

- Modelo: contiene los datos y su estructura. Existen diferentes tipos de elementos QML para crear los modelos.
- Vista: contiene cómo se representan los datos. La vista puede mostrar los datos en forma de lista o de matriz.
- Delegado: establece cómo deben aparecer los datos en la vista. Encapsula los datos del modelo haciendo que sean accesibles a través de él.

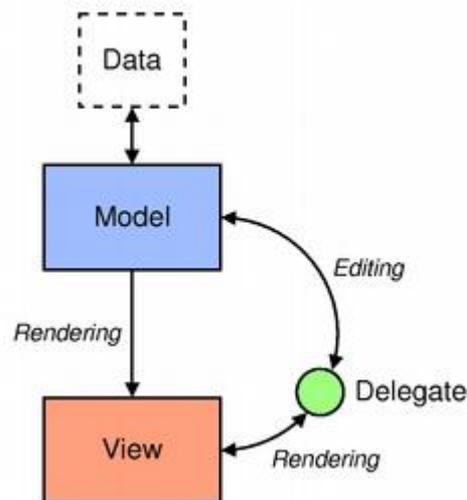


Figura 3 Modelo QtQuick

Esta separación se realiza con el fin de facilitar a los desarrolladores el control sobre las diferentes representaciones de los datos.

Creación interfaces QML

Todas las aplicaciones que desarrollemos basadas en esta tecnología necesitan de diferentes interfaces en QML para funcionar, pero ¿cómo podemos crear dichas interfaces? A la hora de realizar los ficheros que contienen dichas interfaces el programador o diseñador cuenta con dos opciones generales: crear las interfaces a partir de Qt Creator, un IDE que nos proporciona Qt o a través de código fuente. La segunda opción no es una opción completa, dado que se trata más de como instanciar elementos o modificar propiedades de componentes QML ya generados. [12]

Qt Creator

Este IDE sigue la filosofía de otros entornos que existen para otro tipo de interfaces gráficas de usuario. Cuenta con una paleta de componentes posibles a utilizar y que arrastrando y soltándolos en la pantalla principal podemos hacer uso de ellos. Esta opción nos permite ir realizando la interfaz que deseamos de forma gráfica con multitud de posibilidades de personalización. Al acabar, se genera un fichero QML el cual debemos utilizar posteriormente para nuestra aplicación. Como añadido cuenta

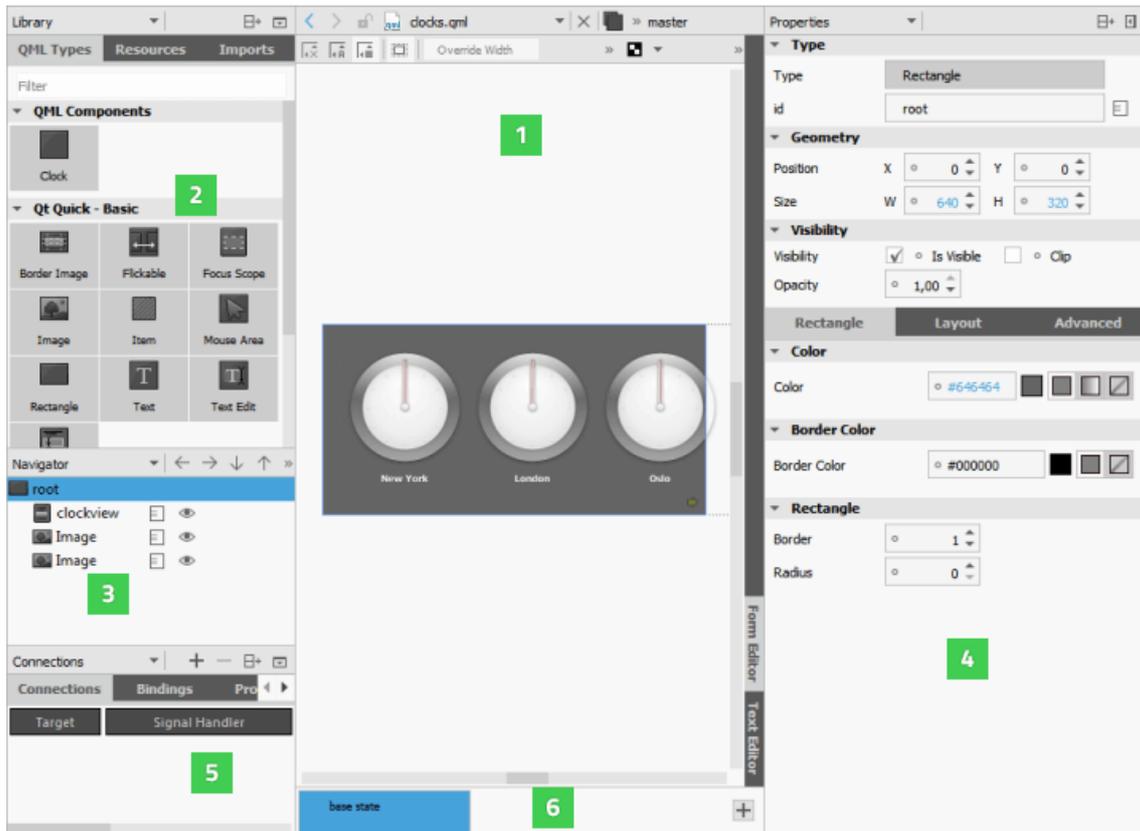


Figura 4. IDE Qt Creator

con una gran variedad de ejemplos ya implementados, así como tutoriales que nos ayudan a aprender cómo se utiliza la aplicación y cómo funcionan las interfaces basadas en QML. [13]

Cuenta con funcionalidades añadidas como integración con sistemas de control de versiones, editor de código fuente QML y C++ o la integración con Visual Studio, entre otras.

Código Fuente

Como se ha comentado previamente esta opción no se puede considerar una opción completa. El motivo de esta consideración viene dado que es necesario disponer previamente de ficheros QML o conocer el lenguaje para poder hacer uso de esta alternativa.

Dado que QML admite JavaScript, así como integración con lenguajes de programación como C++, es posible instanciar y editar objetos QML desde ellos. De esta forma, partiendo de elementos básicos y a través de una serie de llamadas que nos proporciona Qt se pueden llegar a generar interfaces completas. Se pueden encontrar ejemplos en muchos de sus manuales. [14]

Capítulo 3. Estudio Previo

Previamente al desarrollo del proyecto se llevó a cabo, bajo iniciativa del tutor de la empresa colaboradora, un estudio previo con el fin de determinar la viabilidad del presente trabajo, así como la tecnología más apta para su implementación. A continuación, pasamos a comentar brevemente los aspectos que HP determinó relevantes para dicho análisis, así como las conclusiones obtenidas.

Para comenzar, se analizó si existe información suficiente para abordar el desarrollo del proyecto. Hay que destacar que antes de responder a esta cuestión el alcance de la aplicación se ve condicionado a las capacidades y disponibilidad del alumno. Esta consideración hace que la información necesaria para la realización del proyecto dependa del grado en el que el propio alumno desarrolle la aplicación.

Contando con todo, se puede afirmar, que bajo dichas circunstancias existe la información básica y necesaria para desarrollar el proyecto de forma inicial abordando las dudas generales y la funcionalidad básica a implementar. Así mismo se previó, que una vez esté el proyecto comenzado, surgirían dudas sobre la forma óptima de abordar partes de la funcionalidad, que no podemos analizar previamente.

La segunda cuestión a analizar fue la posibilidad de realización del proyecto utilizando las tecnologías solicitadas, en nuestro caso por la empresa. Aunque la respuesta a esta cuestión se realizó de marea teórica, puesto que en el momento del estudio previo el alumno no conocía de manera extensa la tecnología empleada, se pudo afirmar que es posible la realización del proyecto utilizando Qt y QML. La razón radica en que se analizó, de manera superficial, el funcionamiento de dichas tecnologías y la facilidad de utilización de las mismas. Además, se observó, tanto en la documentación oficial como en libros de desarrollo con Qt, la versatilidad y uniformidad de la tecnología, así como sus similitudes con otras tecnologías conocidas. Todo esto hizo que se previera la posibilidad de realización del proyecto utilizando Qt y QML.

En tercer lugar, se determinó el conocimiento de la tecnología solicitada por parte de los desarrolladores. Aunque para esta cuestión la respuesta esperada es una afirmación o una negación, en este caso en concreto es obligatorio dar más detalles. La razón es que dada la variedad de aplicaciones utilizadas a lo largo del estudio del grado en informática se ven diferentes tecnologías de forma aplicada sin conocer o sin entrar en profundidad en la implementación de las mismas. Es decir:

- Se han utilizado programas de terceros que disponían de partes implementadas con tecnología Qt y QML.
- No se ha desarrollado o estudiado la forma de desarrollo basada en esta tecnología.

Es por ello por lo que podemos afirmar que conocemos la tecnología en un producto final, pero se desconoce la forma de realización e implementación de proyectos a partir de dicha tecnología.

Esta situación hace que la parte inicial del desarrollo del proyecto se vea claramente condicionada al aprendizaje de la tecnología, aunque previsto este problema, se ha realizado un pequeño aprendizaje durante el mes previo al comienzo del proyecto con el fin de paliar las consecuencias de abordar un proyecto sin conocimiento de la tecnología a utilizar. [1][2]

La última cuestión planteada fue si sería posible realizar el mismo proyecto más rápido usando otra tecnología conocida por los desarrolladores. Se determinó que existen diferentes tecnologías utilizadas por los desarrolladores para la realización de proyectos similares, y que se han utilizado diferentes lenguajes, como Java o Python, acompañados con las librerías complementarias para el desarrollo de diferentes aplicaciones con interfaz gráfica de usuario. Es por estas razones, por lo que el desarrollo de la parte gráfica y de implementación básica se vería reducida temporalmente en el caso de utilizar una de las tecnologías conocidas previamente.

El problema en ese supuesto, radica a la hora de generar los ficheros en formato QML. Dado que se trata de una restricción del proyecto, aun contando con opciones como PyQt, aumentaría su complejidad de forma considerable, tanto por la capacidad de encontrar guías y referencias de ayuda como la propia codificación del proyecto. Por lo tanto, la conclusión obtenida es que, aun contando con cierta ventaja dado el conocimiento previo de otras tecnologías, el tiempo total de desarrollo del proyecto no se vería reducido. [3]

Conclusiones del estudio. Tecnología elegida.

Una vez considerados todos los puntos hemos determinado que, aun contando con el hándicap de no haber utilizado anteriormente las tecnologías requeridas, la mejor forma de realizar el proyecto es usando dichas tecnologías.

Se prevé que el desarrollo, al tratarse de aplicar nuevos conocimientos, se realice siguiendo una curva de aprendizaje, pero dada la disponibilidad temporal, así como las ganas de aprendizaje, hacen que se afronte el reto de manera optimista.

A esto hay que sumar el hecho de que dichas tecnologías son ampliamente usadas por grandes empresas del sector, por lo que un conocimiento, aunque sea centrado en una aplicación, proporciona una característica muy deseable tanto en el proyecto como en el propio desarrollador, añadiendo atractivo a la elección realizada.

Capítulo 4. Desarrollo del proyecto

4.1. Proceso de desarrollo

Como proceso de desarrollo de software para la realización de este proyecto se decide utilizar el Proceso Unificado Racional o RUP. [15]

El Proceso Unificado Racional se caracteriza por:

- Dirigido por los casos de uso:
- Centrado en la arquitectura
- Iterativo e incremental

Al seguir este proceso podemos enfocarnos en los aspectos fundamentales de un proceso de desarrollo iterativo, como la gestión de los riesgos y la monitorización del progreso. Existen aspectos que este proceso de desarrollo no cubre, como la gestión de personal o la gestión de presupuestos, aspectos importantes en proyectos de mayores dimensiones pero que no poseen demasiada importancia en este proyecto.

Las fases en las que se organiza un proyecto que sigue esta metodología son:

1. Inicio: se establece el ámbito y el alcance del proyecto.
2. Elaboración: se detallan, como mínimo, el 80% de los casos de uso del sistema. Además, se elabora el plan de proyecto a seguir, así como el diseño de la arquitectura.
3. Construcción: se realiza la implementación del proyecto y se detallan los casos de uso restantes de la etapa anterior. Una vez finaliza esta etapa se deben de haber implementado todos los casos de uso del sistema.
4. Transición: se verifica y prueba el funcionamiento de la aplicación realizando, en caso de ser necesario, la corrección de los errores encontrados. Al finalizar esta etapa se proporciona la aplicación al usuario final.

Cada una de estas fases se divide a su vez en iteraciones, de forma que se pueda ir monitorizando el progreso. En la siguiente imagen se puede ver como varía el esfuerzo a realizar dependiendo de la fase en la que se encuentre el proyecto.



Figura 5 Esfuerzo en actividades según la fase. RUP.

4.2. Gestión de riesgos

En este apartado se describen los principales riesgos que pueden afectar al proyecto, así como su posibilidad de ocurrencia y las consecuencias estimadas en el caso de que ocurran. De esta forma podemos prevenir aquellos riesgos con mayor probabilidad de ocurrencia o diseñar medidas de paliar las consecuencias. Para ello vamos a definir para cada riesgo 3 características:

1. Tipo de riesgo
 - a. Riesgos de proyecto: relativos a problemas de coordinación interna del equipo, restricciones de recursos, relación con los proveedores, etc.
 - b. Riesgos de proceso: relativos a un diseño pobre de la aplicación, gestión pobre de los requisitos, mala planificación, etc.
 - c. Riesgos de producto: relativos a la falta de experiencia en el dominio, requisitos incompletos, diseño complejo, etc.

2. Probabilidad de ocurrencia
 - a. Muy improbable: por debajo del 20%
 - b. Improbable: entre el 20% y el 40%
 - c. Posible: entre el 40% y el 60%
 - d. Muy probable: entre el 60 % y el 80%
 - e. Casi seguro: por encima del 80%

3. Impacto en incremento de tiempo
 - a. Insignificante: incremento de menos de un 5% respecto a la estimación inicial.
 - b. Menor: incremento de entre un 5% y un 10% respecto a la estimación inicial.
 - c. Serio: incremento de entre un 10% y un 15% respecto a la estimación inicial.
 - d. Desastroso: incremento de entre un 15% y un 20% respecto a la estimación inicial.
 - e. Catastrófico: incremento superior al 20% respecto a la estimación inicial.

Con estas dos últimas categorías obtenemos una matriz de probabilidad-impacto la cual nos indica que tipo de riesgos son los que más pueden influir en el desarrollo de nuestro proyecto.[16]

Casi seguro	5	5	10	15	20	25
Muy probable	4	4	8	12	16	20
Posible	3	3	6	9	12	15
Improbable	2	2	4	6	8	10
Muy improbable	1	1	2	3	4	5
Probabilidad	X	1	2	3	4	5
Impacto		Insignificante	Menor	Serio	Desastroso	Catastrófico

Figura 6 Estimación del producto <<probabilidad x impacto>> para evaluar riesgos

Esta tabla servirá para estimar qué tratamiento dar a cada riesgo. Los riesgos en la por encima de la línea roja harían que la entrega del proyecto fuese prácticamente inviable mientras que los riesgos de la zona blanca podemos elegir soportarlos. [17]

En las siguientes tablas se establecen alguno de los riesgos con los que nos podemos encontrar a lo largo del proyecto, así como el plan de acción en el caso de ocurrencia.

Riesgo 001		Problemas en la creación de ficheros QML	
Tipo		Producto	
Probabilidad		Posible	
Impacto		Serio	
Descripción		La falta de experiencia con la tecnología QML puede hacer que no sea posible conseguir	
Fase		Construcción	
Estrategia		Investigar el riesgo	
Plan de acción		Realizar investigación y pruebas de creación mediante otras herramientas en etapas previas a la afectada.	

Tabla 2. Riesgo 001 – Problemas en la creación de ficheros QML

Riesgo 002		Modificación de los requisitos	
Tipo		Proceso	
Probabilidad		Muy Improbable	
Impacto		Desastroso	
Descripción		La modificación de los requisitos en etapas posteriores puede provocar el retraso de la entrega del proyecto	
Fase		Elaboración Construcción	
Estrategia		Protegerse del riesgo	
Plan de acción		Invertir el tiempo necesario en realizar el alcance de la aplicación comprobando los resultados con el resto de integrantes del equipo.	

Tabla 3 Riesgo 002 – Modificación de los requisitos

Riesgo 003		Retraso en la planificación	
Tipo		Proceso	
Probabilidad		Posible	
Impacto		Serio	
Descripción		Debido a una estimación de tiempo realizada de forma positiva puede hacer que no se cumplan los plazos y se retrase la fecha de finalización del proyecto.	
Fase		Inicio Elaboración Construcción Transición	
Estrategia		Reservar el riesgo	
Plan de acción		Se establece unan fechas de entrega con holgura suficiente para poder hacer frente a diferentes imprevistos.	

Tabla 4 Riesgo 003 – Retraso en la planificación

Riesgo 004		Problemas con el diseño	
Tipo		Proceso	
Probabilidad		Improbable	
Impacto		Desastroso	
Descripción		Un diseño no completo o un diseño incorrecto de la aplicación provocaría problema en la etapa de construcción haciendo que se tuvieran que modificar los recursos asociados a esa etapa.	
Fase		Elaboración Construcción	
Estrategia		Protegerse del riesgo	
Plan de acción		Invertir el tiempo necesario en la etapa de elaboración y comprobar el resultado con el resto de integrantes del equipo.	

Tabla 5 Riesgo 004 – Problemas con el diseño

Riesgo 005		Problemas con el desarrollo de software	
Tipo		Proceso	
Probabilidad		Muy Improbable	
Impacto		Desastroso	
Descripción		La falta de conocimiento o algoritmos para abordar la implementación del proyecto puede hacer inviable la realización del proyecto	
Fase		Construcción	
Estrategia		Investigar el riesgo	
Plan de acción		Buscar soluciones similares realizadas previamente por el propio equipo o por otros equipos. Investigar los procedimientos de implementación con la tecnología del proyecto.	

Tabla 6. Riesgo 005 – Problemas con el desarrollo de software

Riesgo 006		Problemas con la estación de trabajo	
Tipo		Proyecto	
Probabilidad		Muy Improbable	
Impacto		Catastrófico	
Descripción		El fallo de los equipos con los que se desarrolla el proyecto podría significar la pérdida completa del proyecto	
Fase		Inicio Elaboración Construcción Transición	
Estrategia		Evitar el riesgo	
Plan de acción		Se utiliza un sistema de control de versiones online, de forma que si el equipo de desarrollo fallase el proyecto no se vería afectado.	

Tabla 7 Riesgo 006 – Problemas con la estación de trabajo

4.3. Integrantes y responsabilidades

En este apartado vamos a enumerar a los integrantes del equipo dentro del proyecto, así como sus funciones y responsabilidades dentro del mismo.

Margarita Gonzalo Tasis	
Rol	Responsabilidad
Jefa de proyecto en la Universidad	Administrar el proyecto y a sus integrantes de forma que se alcancen los objetivos del mismo. Solucionar dudas de realización generales sobre el proyecto. Solucionar dudas específicas sobre la implementación del proyecto.

Tabla 8 Integrantes y responsabilidades – Margarita Gonzalo Tasis

Miguel Ángel Benítez Andrades	
Rol	Responsabilidad
Jefe de proyecto en Empresa	Solucionar dudas específicas sobre la implementación del proyecto y la tecnología empleada en el mismo. Comprobación de los resultados del proyecto.

Tabla 9 Integrantes y responsabilidades – Miguel Ángel Andrades

Alberto Gutiérrez Pérez	
Rol	Responsabilidad
Analista	Obtener la especificación de los requisitos y los casos de uso en los que se basa el proyecto.
Diseñador	Realizar el diseño del proyecto con el fin de cumplir los objetivos y los requisitos utilizando patrones conocidos.
Desarrollador	Realizar la implementación del proyecto
Tester	Comprobar y verificar la no existencia de errores o fallos en la aplicación.

Tabla 10 Integrantes y responsabilidades – Alberto Gutiérrez Pérez

4.2. Planificación Temporal

La realización del proyecto se ha realizado siguiendo la planificación desarrollada, de forma que cada uno de los hitos que se enumeran en la misma representan versiones funcionales e incompletas de la aplicación. Posteriormente se enumerarán de forma superficial las características que se encontrarán implementadas en cada uno de los hitos proporcionados.

En las siguientes páginas se muestran la planificación realizada de dos formas diferentes. En la primera imagen se muestran todas las tareas en forma de lista con sus detalles. De esta forma se puede examinar cuando empieza cada una de las tareas, así como la duración estimada de las mismas.

En una segunda imagen se puede observar la secuenciación de las actividades de forma gráfica, de forma que sea comprensible de forma intuitiva. Hay que destacar que para cualquier tipo de consulta sobre las fechas o detalles de las tareas esta imagen no es válida, esto es debido a que, aunque se muestren exactamente los mismos datos, la exportación de los mismos al formato gráfico hace que el tiempo de la planificación se exprese en intervalos de varios días.

Hito	Descripción	Fecha
Hito 1	Interfaces iniciales de la aplicación y navegación básica	24/02/2017
Hito 2	Creación inicial de ventanas de tipo HomeScreen	17/03/2017
Hito 3	Creación inicial de ventanas de tipo Información	29/03/2017
Hito 4	Creación inicial de ventanas de tipo nivel de tinta	15/04/2017
Hito 5	Modificación y corrección de la creación de ventanas de tipo HomeScreen e información	26/04/2017
Hito 6	Modificación y corrección de la creación de ventanas de tipo niveles de tinta y de la opción editar.	12/05/2017

Tabla 11. Planificación temporal

4.3. Vista en detalle

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1		Estudio Previo	6 días	lun 16/01/17	sáb 21/01/17	
2		Analisis	9 días	lun 23/01/17	mié 01/02/17	1
3		Diseño	7 días	jue 02/02/17	jue 09/02/17	2
4		Implementación	56 días	vie 10/02/17	sáb 15/04/17	3
5		Diseño Interfaces	12 días	vie 10/02/17	jue 23/02/17	
6		Navegación	1 día	vie 24/02/17	vie 24/02/17	5
7		Hito 1	0 días	vie 24/02/17	vie 24/02/17	6
8		Creación ventanas tipo Home Screen	18 días	sáb 25/02/17	vie 17/03/17	6
9		Hito 2	0 días	vie 17/03/17	vie 17/03/17	8
10		Creación ventanas tipo Información	10 días	sáb 18/03/17	mié 29/03/17	8
11		Hito 3	0 días	mié 29/03/17	mié 29/03/17	10
12		Creación de ventanas tipo niveles tinta	15 días	jue 30/03/17	sáb 15/04/17	10
13		Hito 4	0 días	sáb 15/04/17	sáb 15/04/17	12
14		Pruebas	23 días	lun 17/04/17	vie 12/05/17	4
15		Pruebas ventanas tipo menu	6 días	lun 17/04/17	sáb 22/04/17	
16		Pruebas ventanas tipo información	3 días	lun 24/04/17	mié 26/04/17	15
17		Hito 5	0 días	mié 26/04/17	mié 26/04/17	16
18		Purbeas ventanas tipo niveles tinta	4 días	jue 27/04/17	lun 01/05/17	16
19		Pruebas editar Interfaces	10 días	mar 02/05/17	vie 12/05/17	18
20		Hito 6	0 días	vie 12/05/17	vie 12/05/17	19
21		Elaboración Documentación	95 días	vie 10/02/17	mié 31/05/17	3

Figura 7. Planificación tareas del proyecto en detalle

4.4. Vista gráfica

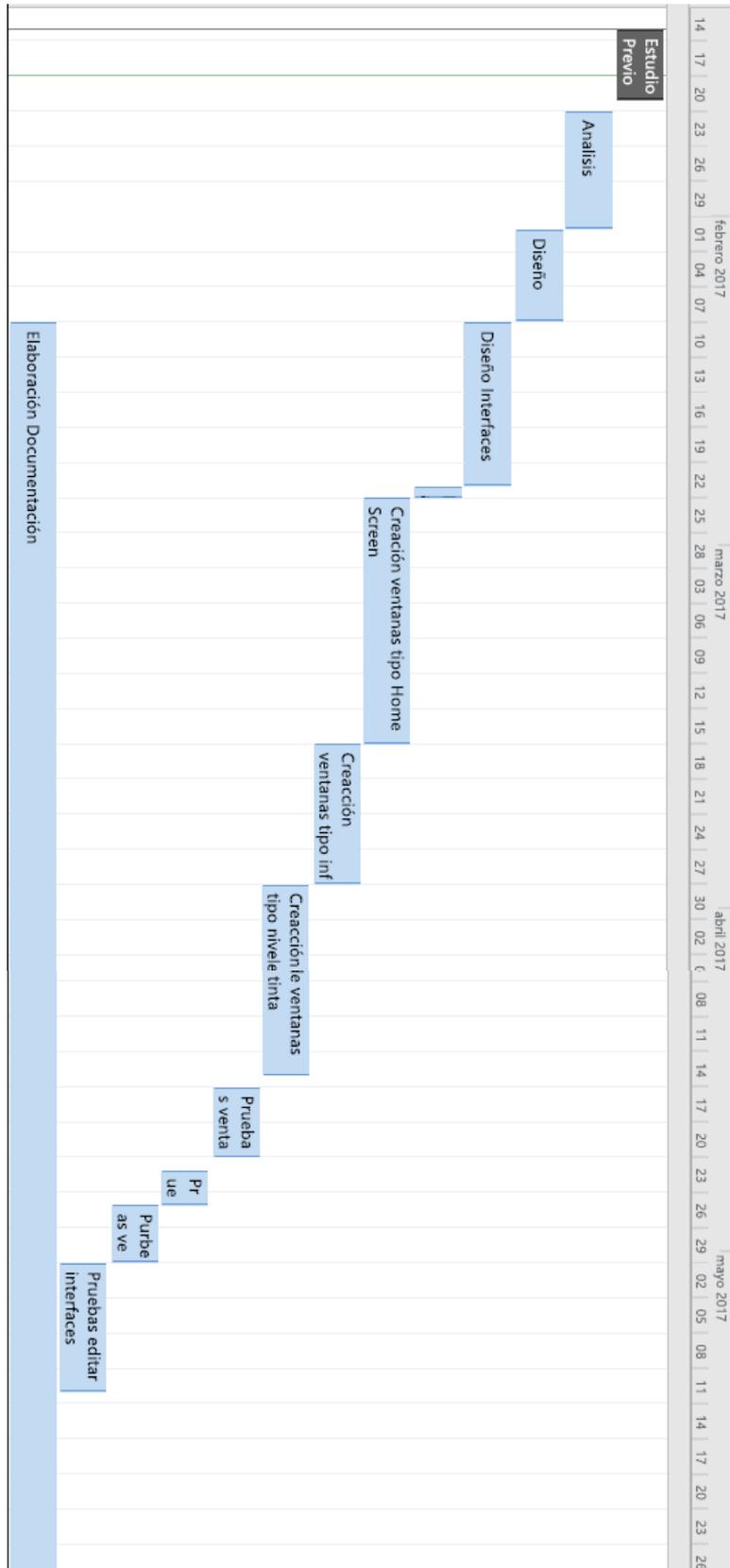


Figura 8 Planificación tareas del proyecto de forma gráfica.

4.5. Seguimiento del proyecto

Este proyecto posee una peculiaridad que ha condicionado su seguimiento. Se trata de un proyecto conjunto en la que se cuenta con dos tutores, uno de los cuales era inaccesible de forma presencial.

Dadas estas características la realización del seguimiento del proyecto se ha llevado a cabo siguiendo la planificación. Al finalizar cada hito se realizaba una entrega en la que se adjuntaban los documentos y ficheros que componen el proyecto, así como un pequeño resumen que indicaba lo realizado en durante ese periodo. Estos resúmenes de cada periodo de desarrollo se encuentran en el [Anexo I](#).

En el caso de sufrir retrasos debido a complicaciones o imprevistos, se adjuntaba el motivo del retraso, así como las investigaciones o posibles soluciones sobre el mismo. Por cada problema encontrado se ha tratado de realizar una estimación temporal de la realización de los arreglos necesarios siguiendo siempre la planificación inicial.

De esta forma los dos tutores involucrados han podido revisar el avance realizado, la realización del proyecto y comentar los problemas encontrados. Como consecuencia de seguir este método de seguimiento el alumno ha realizado un desarrollo continuo a lo largo de todo el proyecto.

La realización del proyecto se planifica para un total de 130 días, trabajando de forma continua 6 días a la semana, con una media de 5 horas al día. Esta planificación se ve alterada dadas las complicaciones y mejoras introducidas dentro del proyecto. Finalmente, el proyecto se realiza en un total de 160 días, alargando el desarrollo aproximadamente un mes respecto de la planificación inicial.

Capítulo 5. Análisis

En el siguiente apartado vamos a presentar el resultado del trabajo de análisis realizado sobre el sistema. Dicho análisis se encuentra formado por diferentes partes, las cuales son:

- Requisitos funcionales
- Requisitos no funcionales
- Actores
- Casos de uso
- Diagrama de secuencia
- Modelo de dominio

Se debe destacar que el análisis se realiza a partir de una breve especificación proporcionada sobre el proyecto. Esta escasa especificación se elabora de forma intencionada, de forma que se establezca el límite mínimo de implementaciones a cumplir una vez finalice el proyecto. A partir de ese límite, está en manos del desarrollador del proyecto establecer el alcance máximo de la aplicación.

5.1. Requisitos Funcionales

Los requisitos funcionales constituyen el comportamiento de la aplicación, las funcionalidades que el sistema debe cumplir. [18]

En las siguientes tablas se establecen los requisitos funcionales de esta aplicación:

RF – 001	Creación Interfaz
Descripción	La aplicación deberá permitir la creación de interfaces

Tabla 12. RF – 001 Creación interfaz

RF – 002	Modificación de interfaces
Descripción	La aplicación deberá permitir la edición de interfaces

Tabla 13. RF – 002 Modificación de interfaces

RF – 003	Tipos de interfaz
Descripción	La aplicación deberá permitir diferentes tipos de interfaces

Tabla 14. RF – 003 Tipos de interfaz

RF – 004	Introducción de parámetros
Descripción	La aplicación deberá permitir la parametrización de las características de la interfaz

Tabla 15. RF – 004 Introducción de parámetros

5.2. Requisitos no funcionales

Los requisitos no funcionales restringen y detallan características de la aplicación, completando a los requisitos funcionales. [18]

RNF – 001	Formato QML
Descripción	La aplicación deberá general ficheros en formato QML

Tabla 16. RNF – 001 Formato QML

RNF – 002	Estilo Asistente
Descripción	La aplicación deberá realizarse al estilo asistente de instalación

Tabla 17. RNF – 002 Estilo Asistente

5.3. Actores

Un actor especifica un rol jugado por un usuario o cualquier otro sistema que interactúe con la aplicación. [18]

En el caso de esta aplicación, el sistema cuenta únicamente con un actor, el programador de interfaces.

Actor- 001	Usuario – Programador
Descripción	Representa a un programador que hace uso del sistema desarrollado.

Tabla 18 Actor 001 – Usuario Programador

5.4. Casos de Uso

Un caso de uso describe la interacción que se debe llevar a cabo, entre el actor y el sistema, con el fin de realizar una tarea. [18]

En la siguiente imagen se muestra en el diagrama el conjunto de casos de uso de la aplicación a desarrollar.

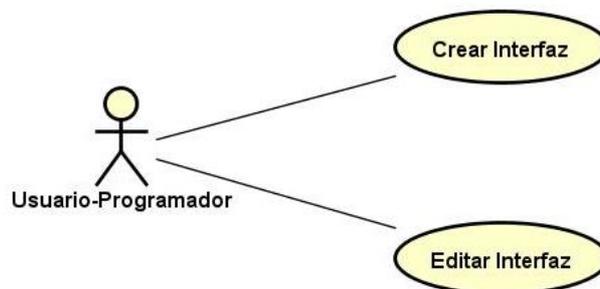


Figura 9. Diagrama de casos de uso

ITEM	VALUE
UseCase	Crear Interfaz
Summary	
Actor	Usuario-Programador
Precondition	
Postcondition	El sistema ha de crear un fichero qml que contenga la interfaz.
Base Sequence	<ol style="list-style-type: none"> 1.- El usuario selecciona que desea crear una interfaz 2.- El sistema le indica los tipos de interfaces disponibles 3.- El usuario selecciona un tipo de interfaz 4.- El sistema muestra las opciones de configuración de la interfaz 5.- El usuario introduce las opciones de configuración que desea personalizar 6.- El sistema guarda la interfaz en un fichero qml y el caso de uso finaliza
Branch Sequence	1',3',5'.- El usuario cancela, el caso de uso queda sin efecto
Exception Sequence	
Sub UseCase	
Note	

Figura 10. CU 1 – Crear Interfaz

ITEM	VALUE
UseCase	Editar Interfaz
Summary	
Actor	Usuario-Programador
Precondition	
Postcondition	
Base Sequence	<p>1.- El usuario selecciona que desea editar una interfaz</p> <p>2.- El sistema le indica la ruta al fichero .qml que contiene la interfaz a editar</p> <p>3.- El usuario introduce la ruta del fichero</p> <p>4.- El sistema comprueba que se trata de un fichero .qml correcto y muestra las opciones de configuración de la interfaz</p> <p>5.- El usuario introduce las opciones de configuración que desea personalizar</p> <p>6.- El sistema guarda la interfaz en un fichero qml y el caso de uso finaliza</p>
Branch Sequence	<p>1',3',5".- El usuario cancela, el caso de uso queda sin efecto</p> <p>3".- La ruta introducida no corresponde con un fichero .qml válido, el sistema muestra un mensaje de error, el caso de uso continúa en el paso 2.</p> <p>3"". - La ruta introducida no existe, el sistema muestra un mensaje de error, el caso de uso continúa en el paso 2.</p>
Exception Sequence	
Sub UseCase	
Note	

Figura 11. CU 2 – Editar Interfaz

Diagramas de secuencia

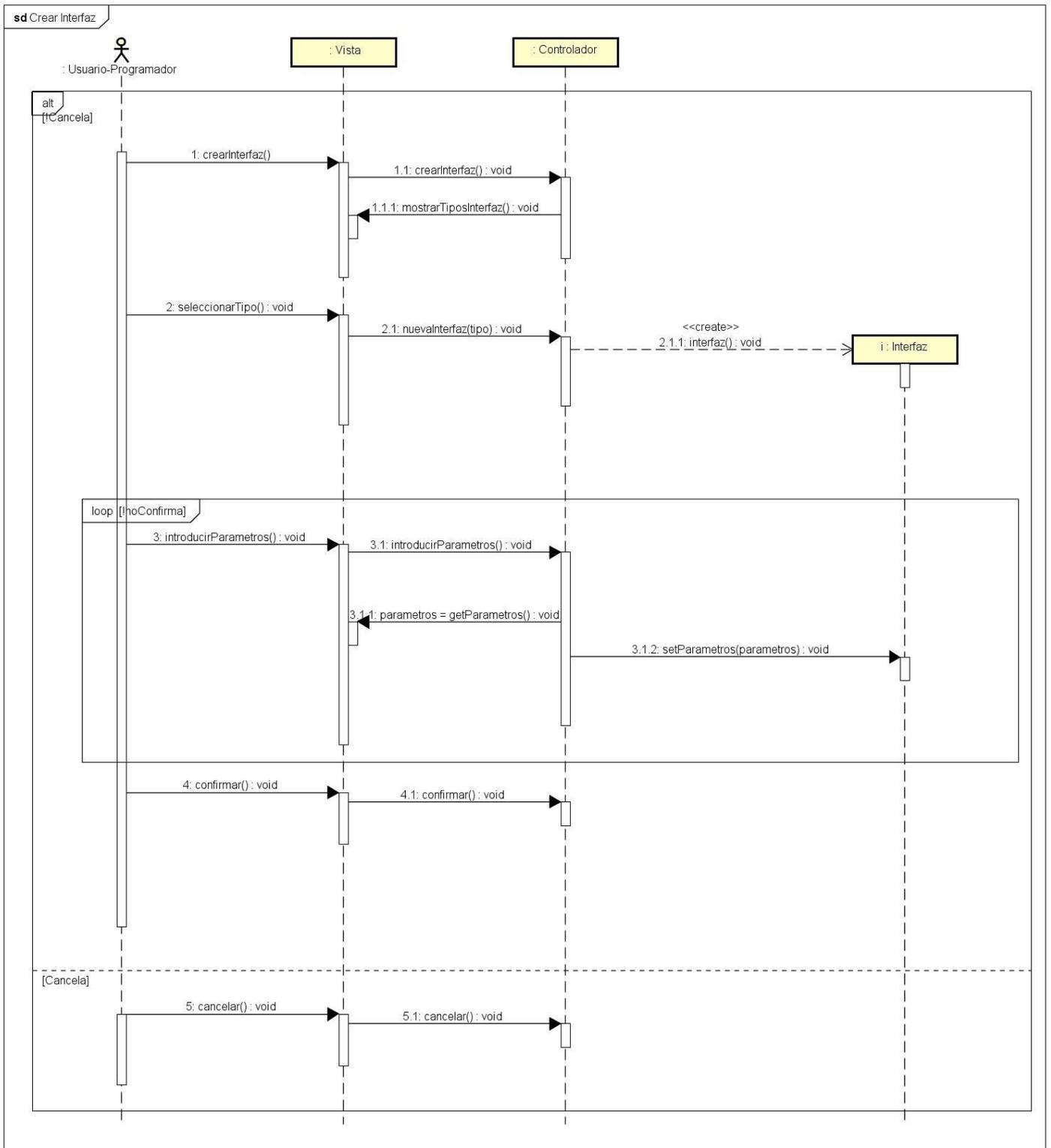


Figura 12. Diagrama de secuencia crear interfaz

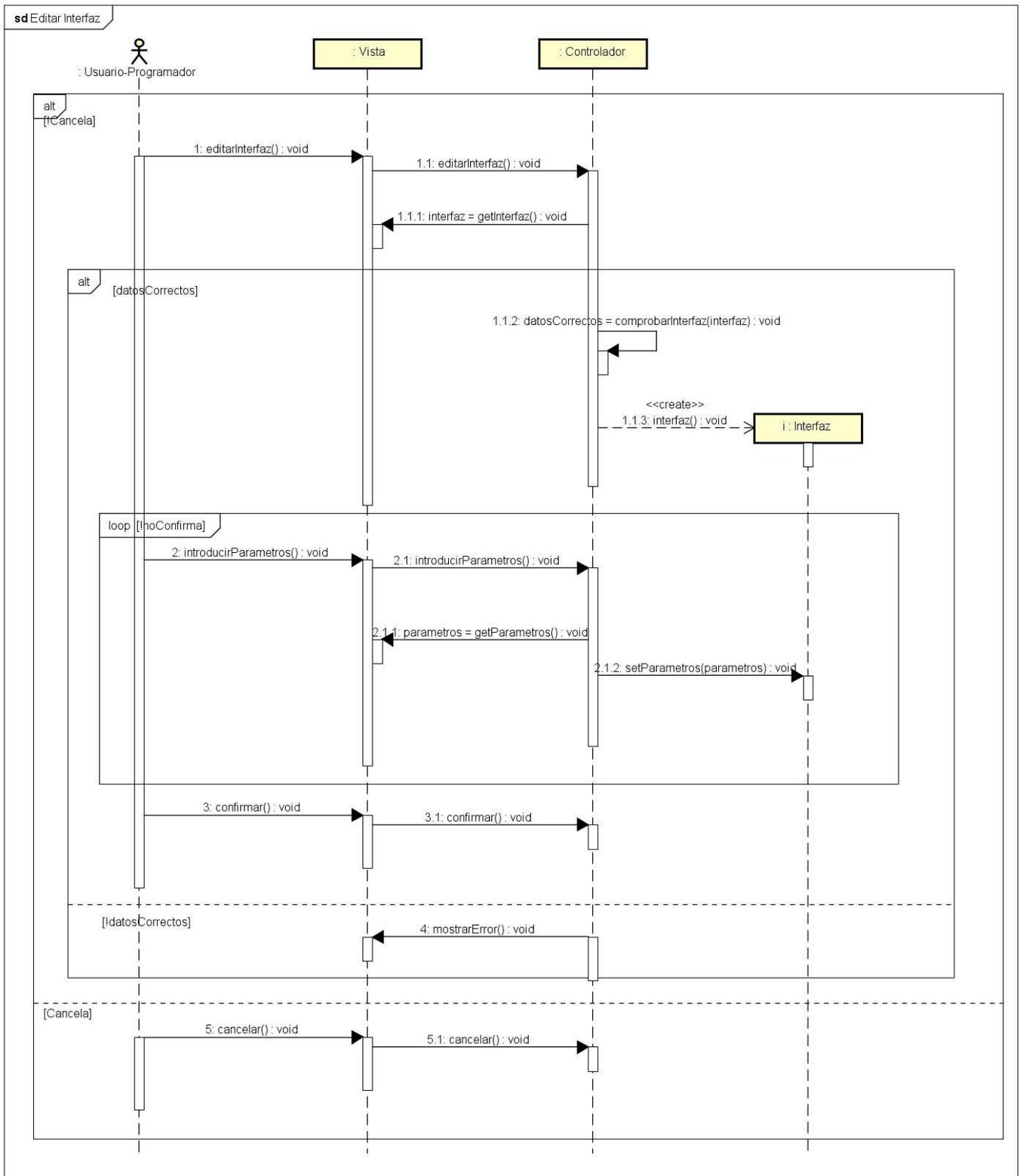


Figura 13. Diagrama de secuencia editar interfaz.

5.5. Modelo de dominio

Este apartado contiene el diagrama de clases de análisis de la aplicación. En el diagrama se pueden ver las clases que componen el sistema, así como sus operaciones básicas. Posteriormente, en la etapa de diseño se precisará y se afinará el diagrama mostrando la totalidad de las operaciones y relaciones.

Como se puede ver en la siguiente imagen, el modelo cuenta con 6 elementos distribuidos de la siguiente forma:

- **Vista:** corresponde a la vista en el patrón arquitectónico Modelo-Vista-Controlador. Se trata de una interfaz de usuario genérica a partir de la cual el usuario interactúa con la aplicación. Gracias a la vista el usuario es capaz de introducir los parámetros de la interfaz QML a generar. En este punto no se cuenta con un diseño gráfico de la misma, aunque se cuenta con algunos prototipos proporcionados por la empresa HP.
- **Controlador:** corresponde al controlador en el patrón arquitectónico Modelo-Vista-Controlador. Sus funciones básicas son la realización de comprobaciones de los elementos introducidos por el usuario y gestionar el flujo de la aplicación. Dentro de la funcionalidad de la aplicación las operaciones de creación y edición de interfaces cobran una vital importancia. El controlador es el encargado de realizar la transformación de los elementos del modelo a ficheros QML válidos y viceversa. Dada esta alta carga de trabajo sobre un único elemento se plantea, para etapas posteriores, dividir las tareas de control puro y las relativas a creación y edición en dos elementos diferentes. De esta forma se mantendrá una separación clara entre funcionalidades de la aplicación.
- **Interfaz:** corresponde al modelo en el patrón arquitectónico Modelo-Vista-Controlador. En ella se albergan los atributos y operaciones comunes a todos los tipos de interfaces. Esta decisión se produce dado que todas las interfaces van a contar con campos como su dimensión o el número de elementos que albergan. Como se puede ver en la imagen, cuenta con un único nivel de herencia, en el que se encuentran tres elementos: Información, Niveles de tinta, Home Screen. Cada uno de estos elementos hace referencia a los diferentes tipos de interfaces que el asistente va a crear.

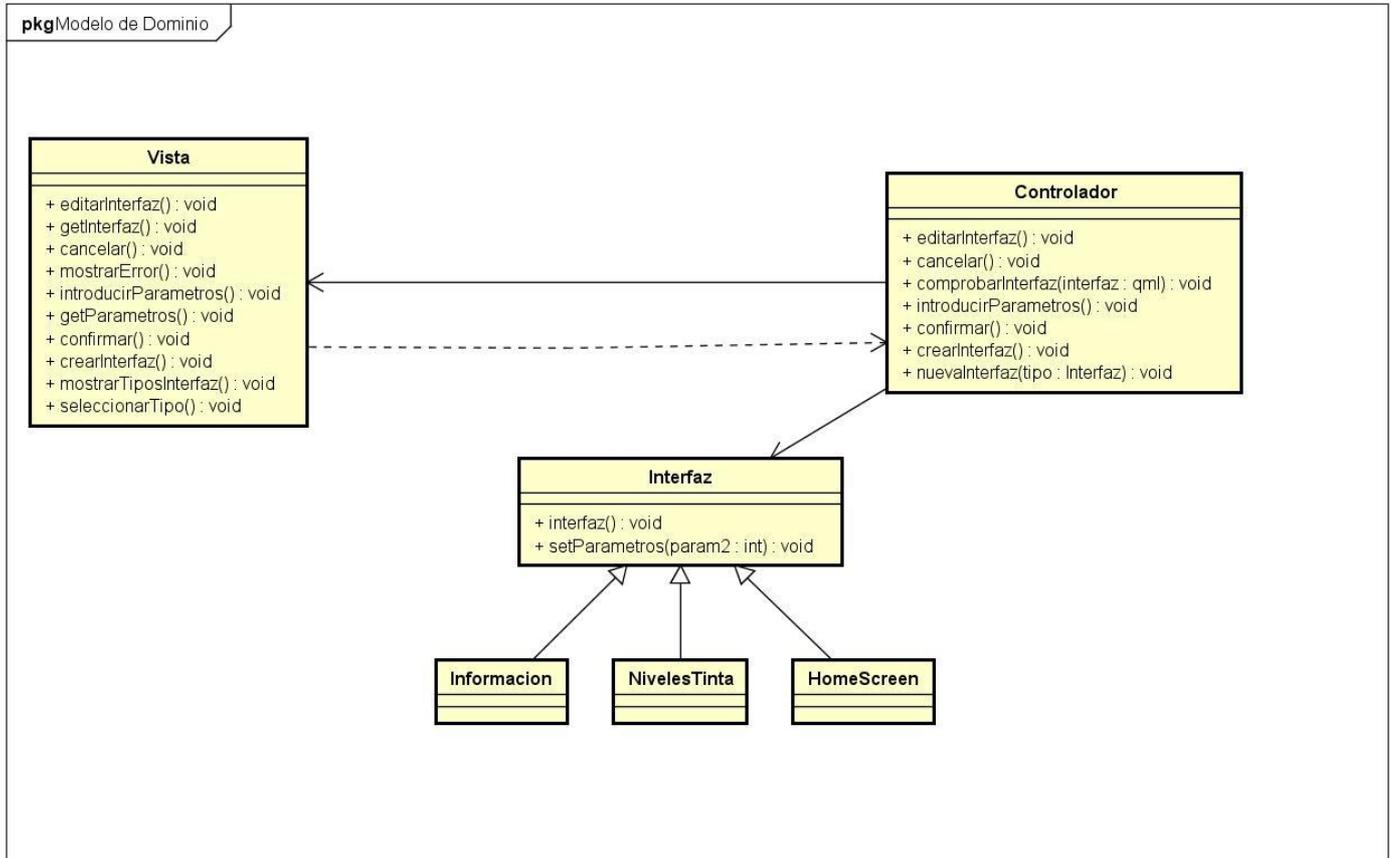


Figura 14. Diagrama modelo de dominio.

Capítulo 6. Diseño

6.2. Arquitectura General

Para la realización del sistema se ha seguido el patrón arquitectónico Modelo-Vista-Controlador. De esta forma mantenemos separados los datos y la lógica de negocio. Por un lado, se encuentran las vistas, las cuales son utilizadas por el usuario con el fin de introducir los parámetros personalizables de cada interfaz a generar.

Por otra parte, se encuentra en modelo, el cual almacena la representación de los datos que posteriormente forman las interfaces a generar. Para ello se cuenta con una interfaz genérica que almacena datos genéricos como el tamaño de la misma, el número de elementos, la disposición de los mismos, etcétera. En un nivel inferior de herencia se encuentran los diferentes tipos de interfaz dentro de los cuales se almacena la información personalizable de cada tipo específico.

El flujo de la aplicación y las comprobaciones de los datos se realizan en la parte correspondiente al controlador. Dentro de este paquete se cuenta con un controlador por cada proceso completo. Es decir, para la creación de una interfaz QML de tipo Menú, existe un controlador, para la creación de una interfaz de tipo Información existe otro controlador, etc. De esta forma cada proceso de la aplicación se encuentra aislado del resto. Debido a que existen ciertas operaciones comunes a todos los procesos existe un controlador genérico del cual heredan el resto.

La aplicación cuenta con paquete a mayores, el paquete Servicios. Este paquete se encarga de realizar la traducción entre el modelo y el fichero QML y viceversa. En una primera aproximación, esta funcionalidad se encontraba integrada dentro de la parte correspondiente al controlador. La decisión de separarlas se toma debido a dos razones principales:

- Mantener aisladas las diferentes funcionalidades del sistema, conservando así la modularidad del mismo.
- La complejidad adquirida por el controlador.

Este paquete se encuentra formado por dos elementos: el primero se encarga de realizar la creación del fichero QML a partir del modelo, el segundo realiza el proceso contrario, a partir de un fichero QML obtiene el modelo correspondiente.

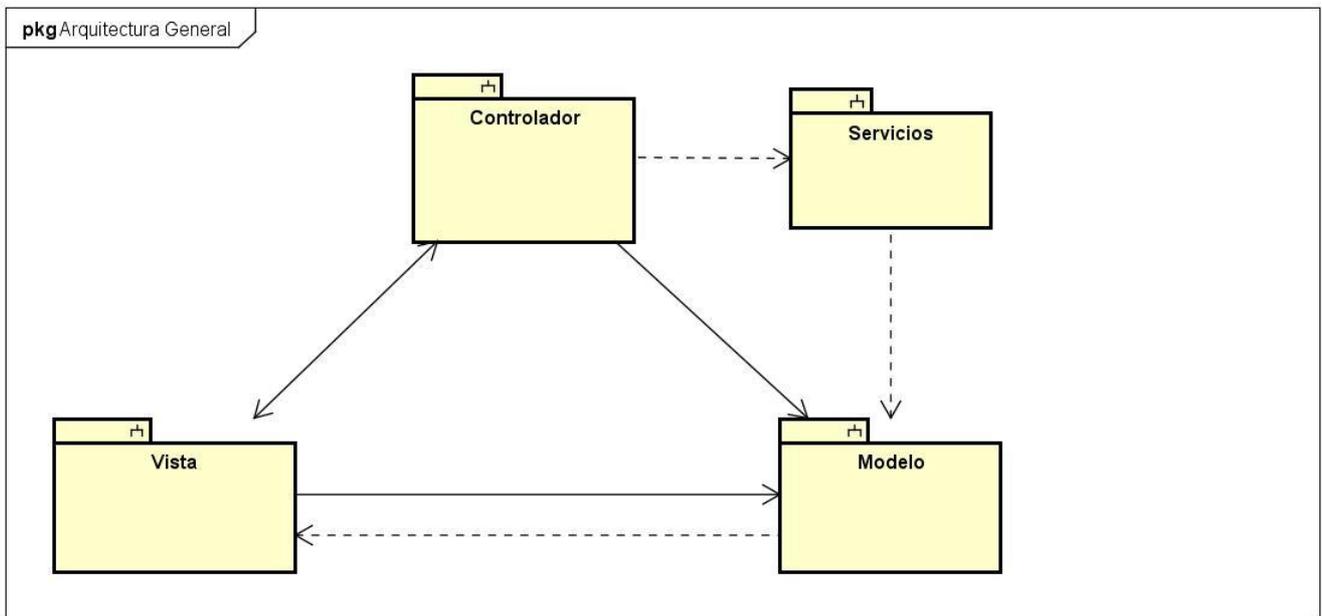


Figura 15. Arquitectura General

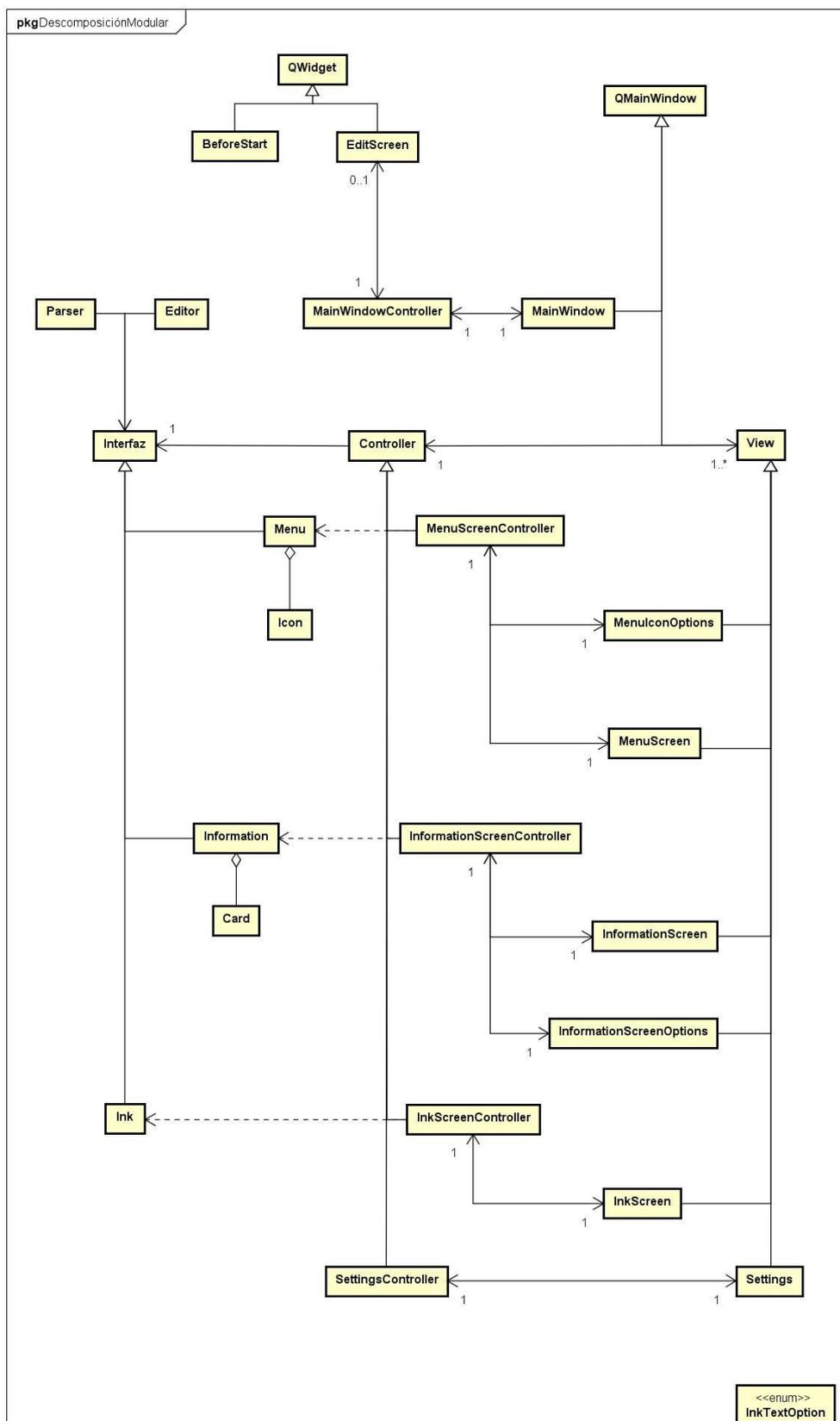


Figura 16. Modelo en dise1o

6.3. Diagramas de secuencia

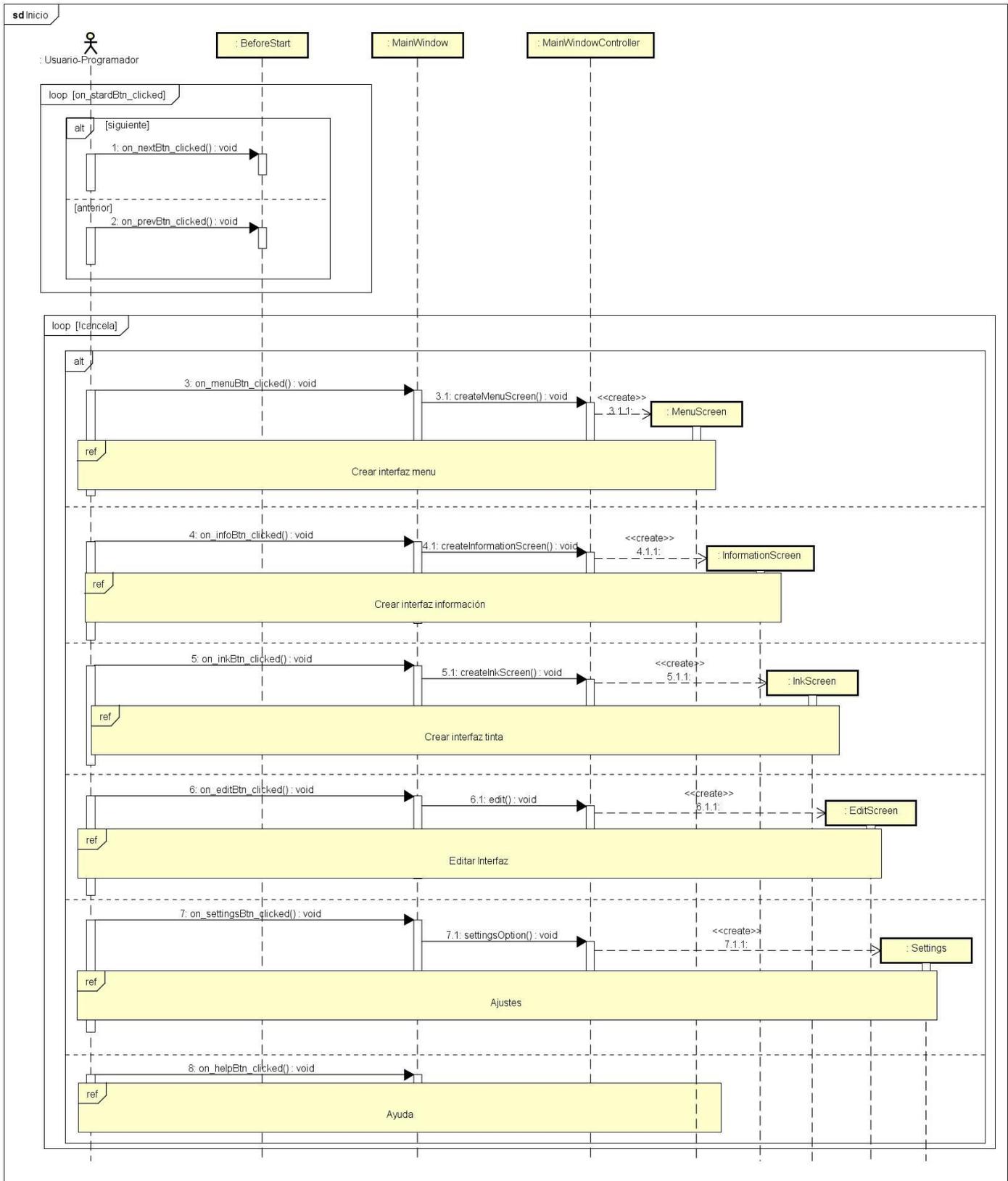


Figura 17. Diagrama de secuencia inicio

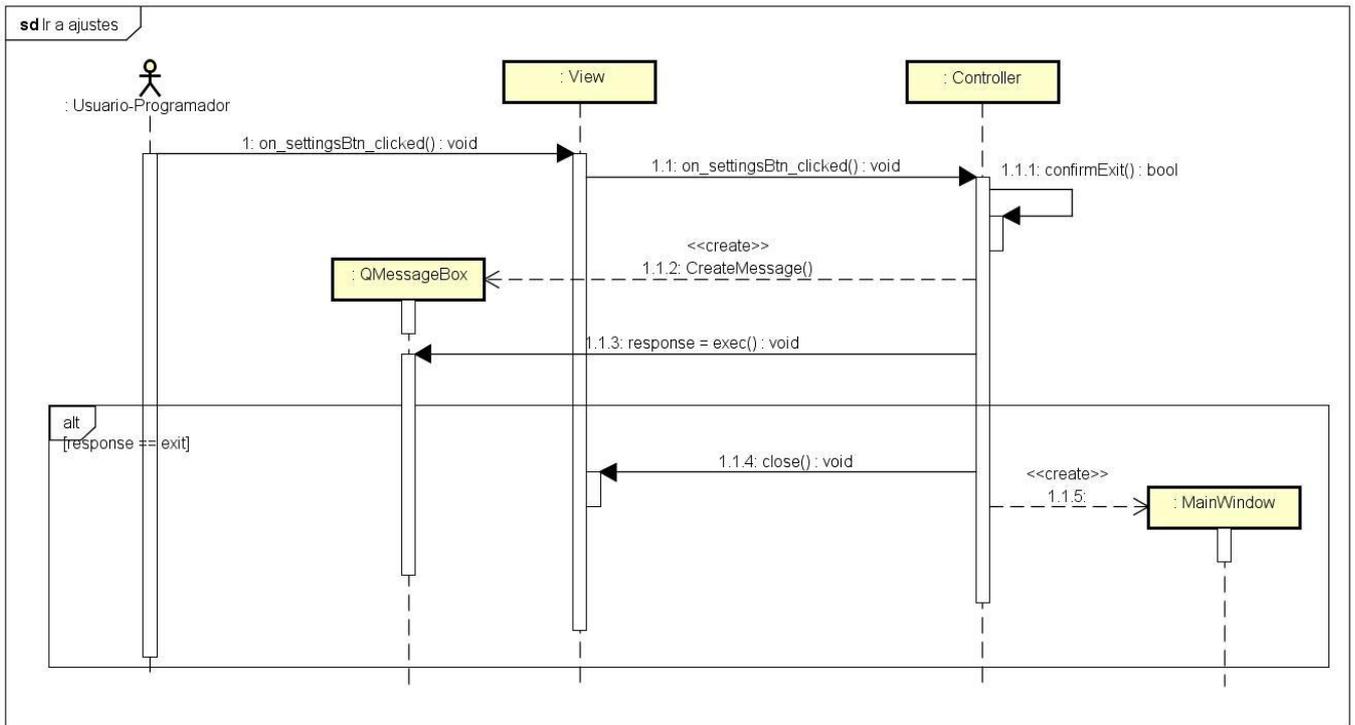


Figura 18. Diagrama de secuencia Ir a Ajustes

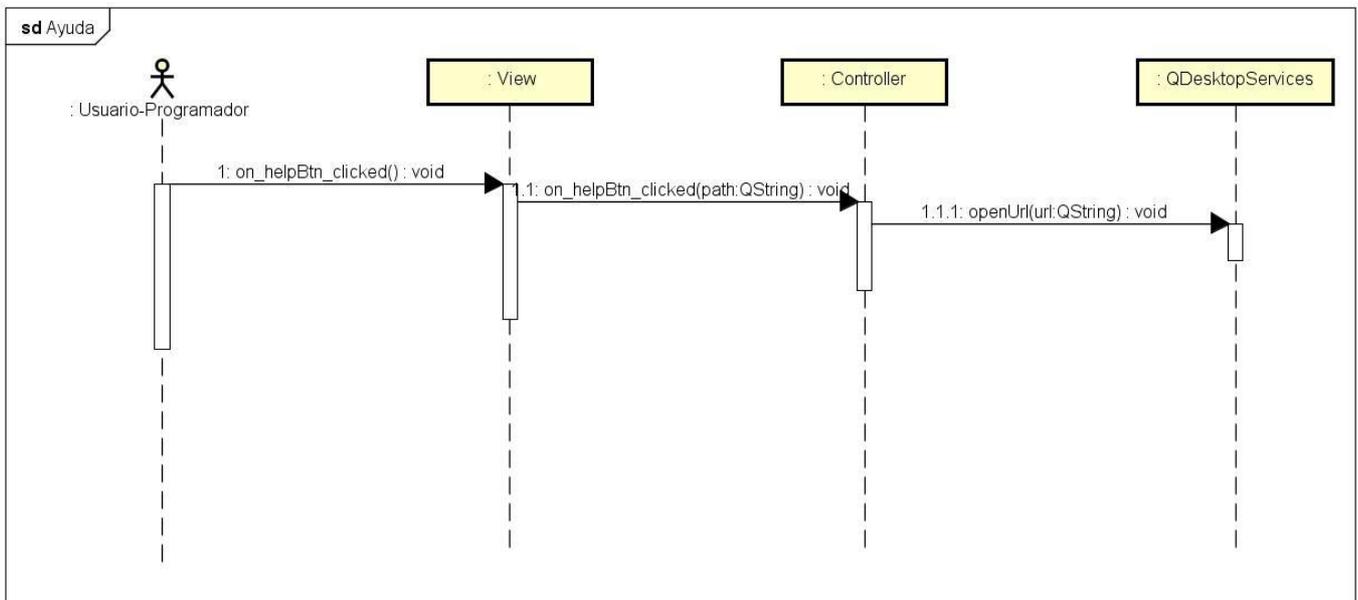


Figura 19. Diagrama de secuencia ayuda

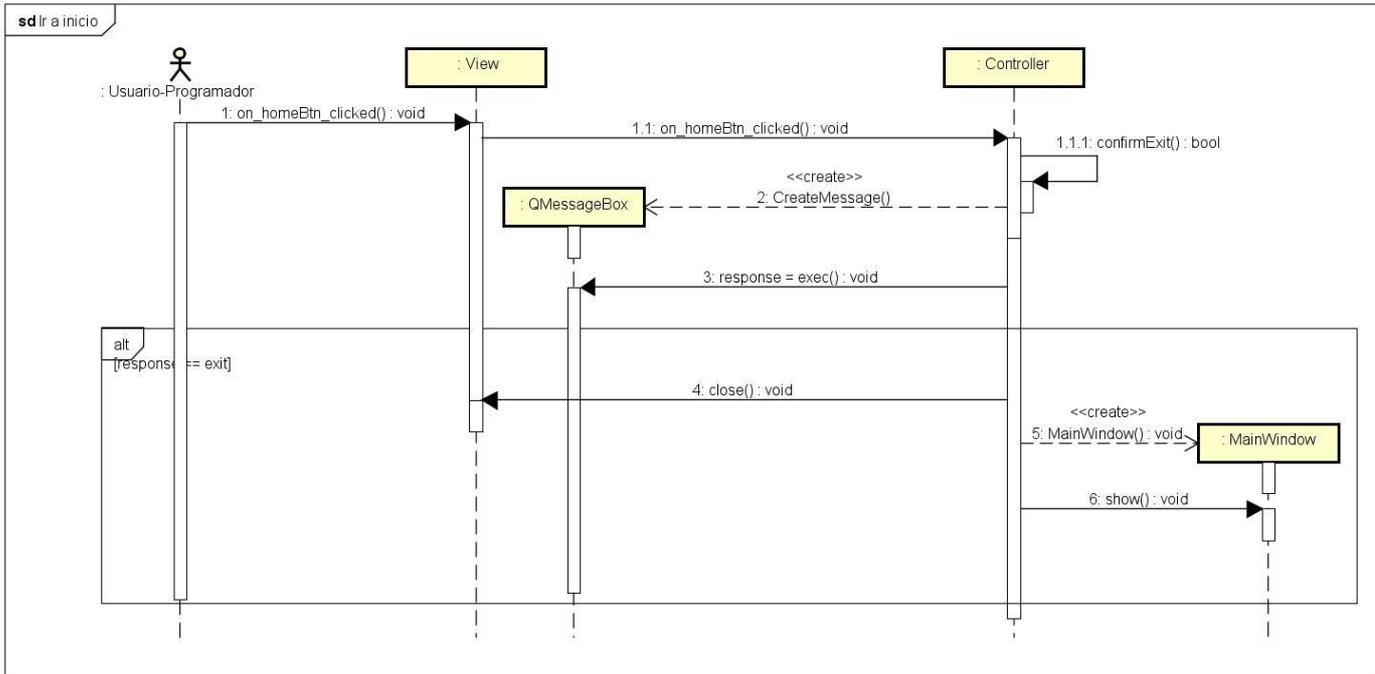


Figura 20. Diagrama de secuencia Ir a inicio

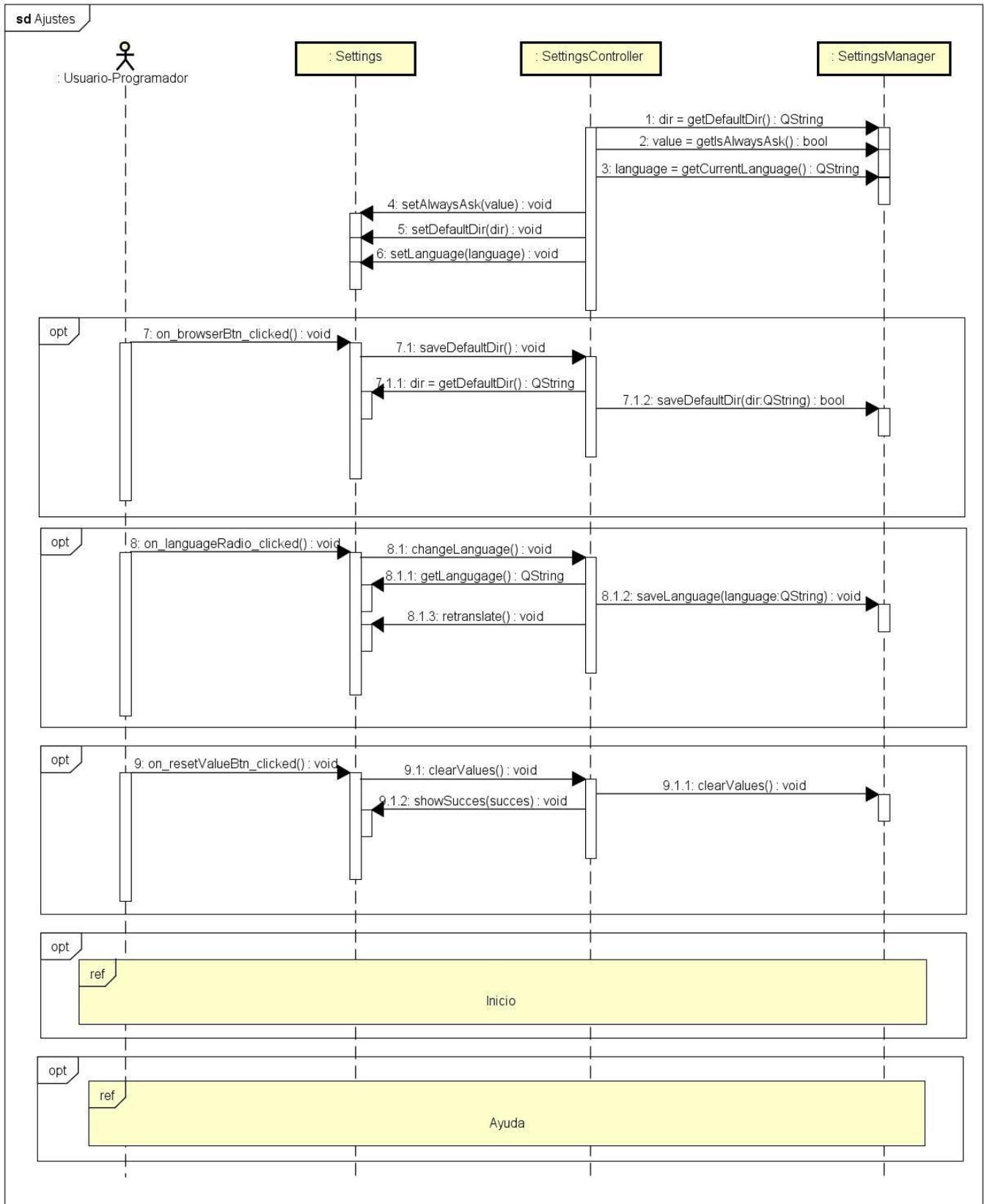


Figura 21. Diagrama de secuencia Ajustes

Creación interfaz tipo Menú

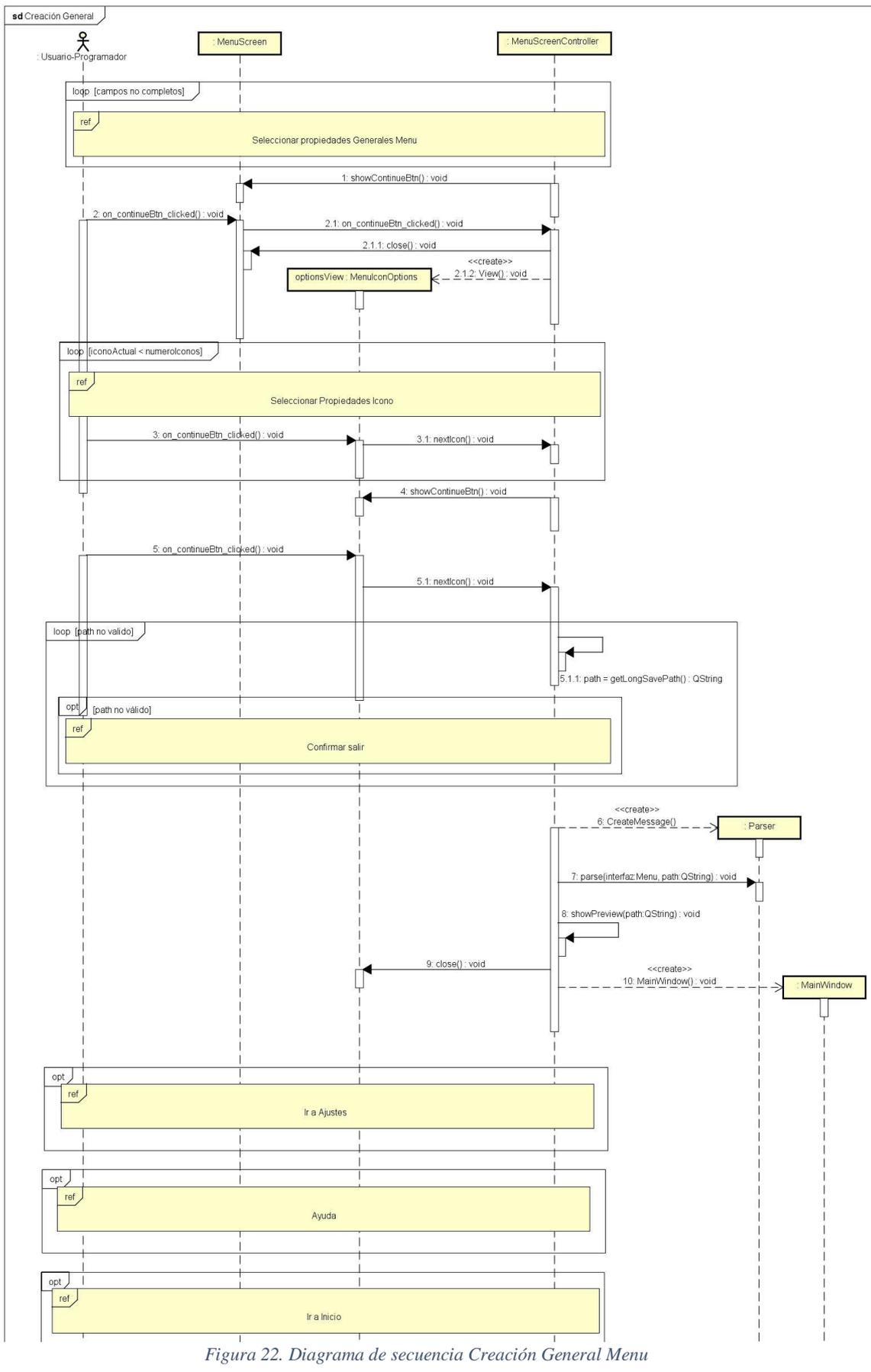


Figura 22. Diagrama de secuencia Creación General Menu

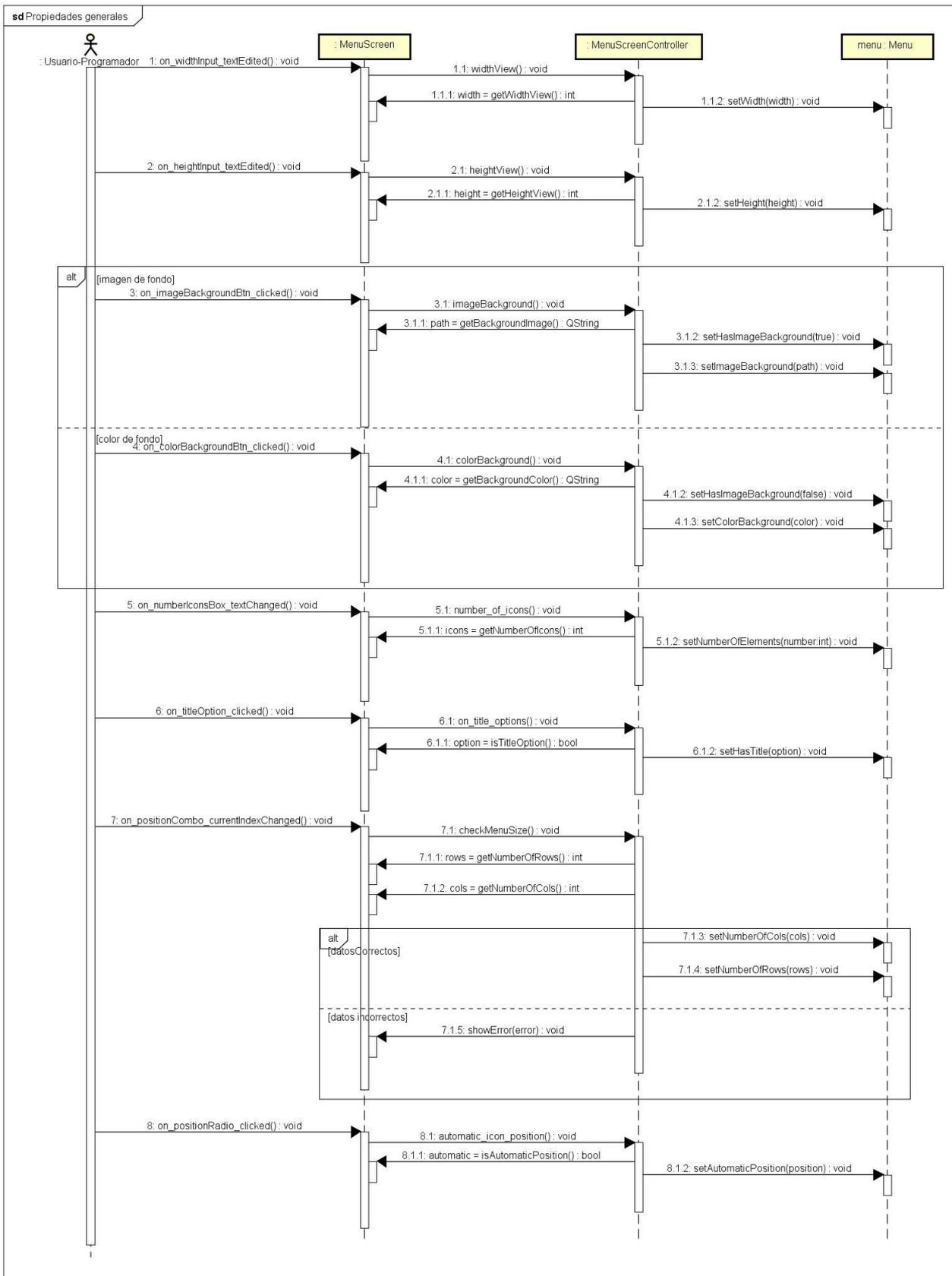


Figura 23. Diagrama de secuencia Propiedades Generales Menú

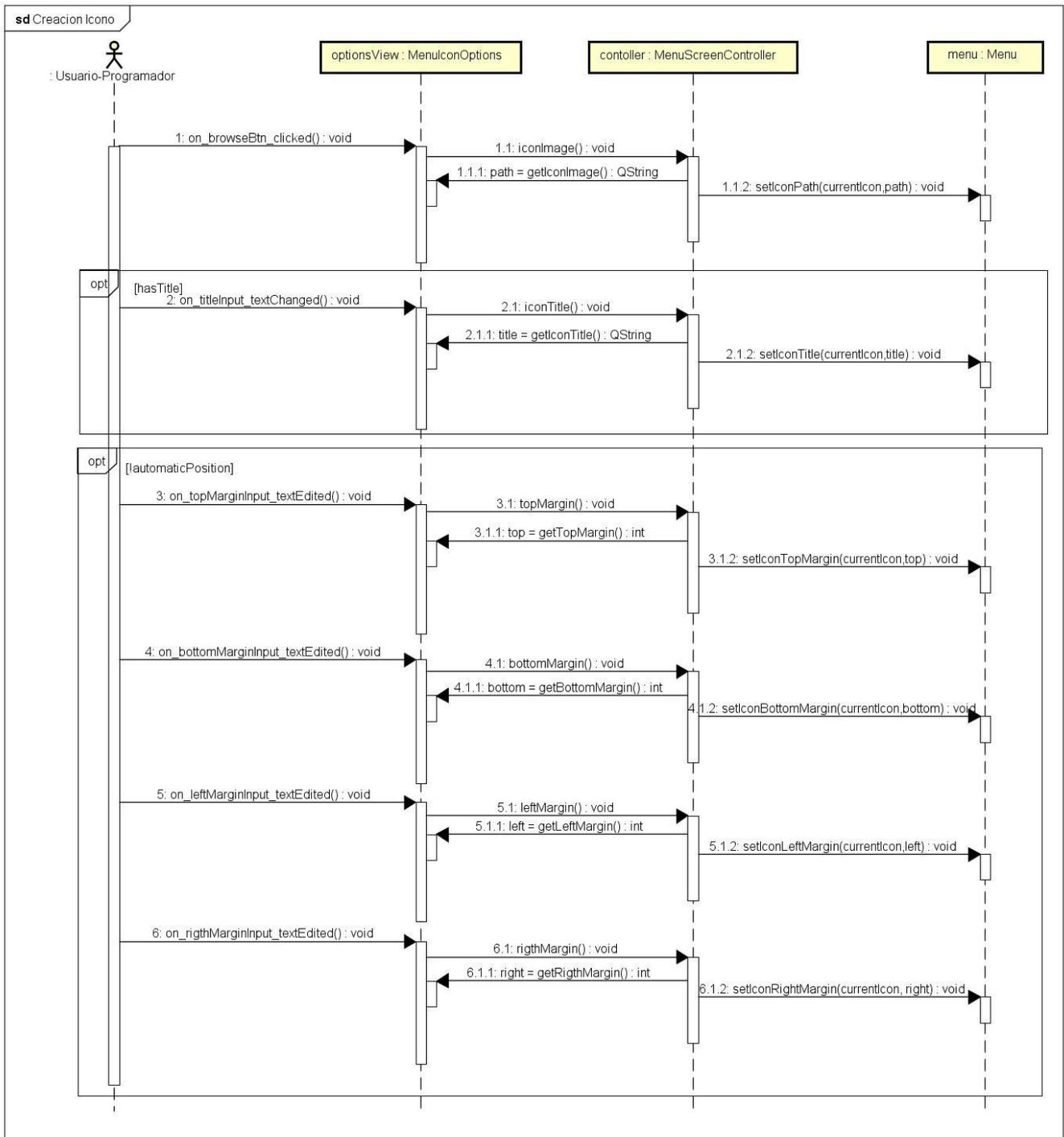


Figura 24 Diagrama de secuencia Creación Icono Menú

Creación interfaz tipo Información

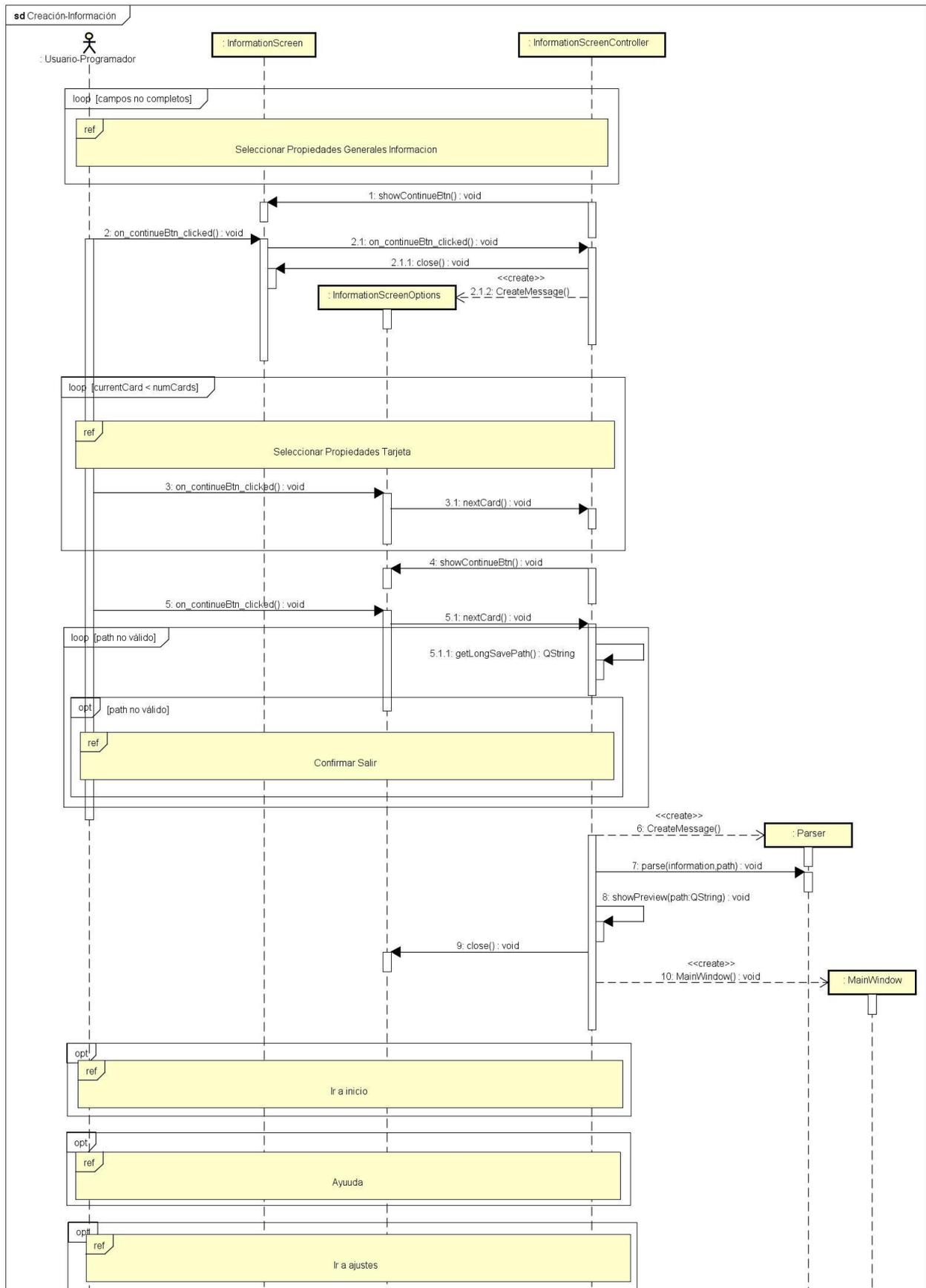


Figura 25 Diagrama de secuencia Creación General Información

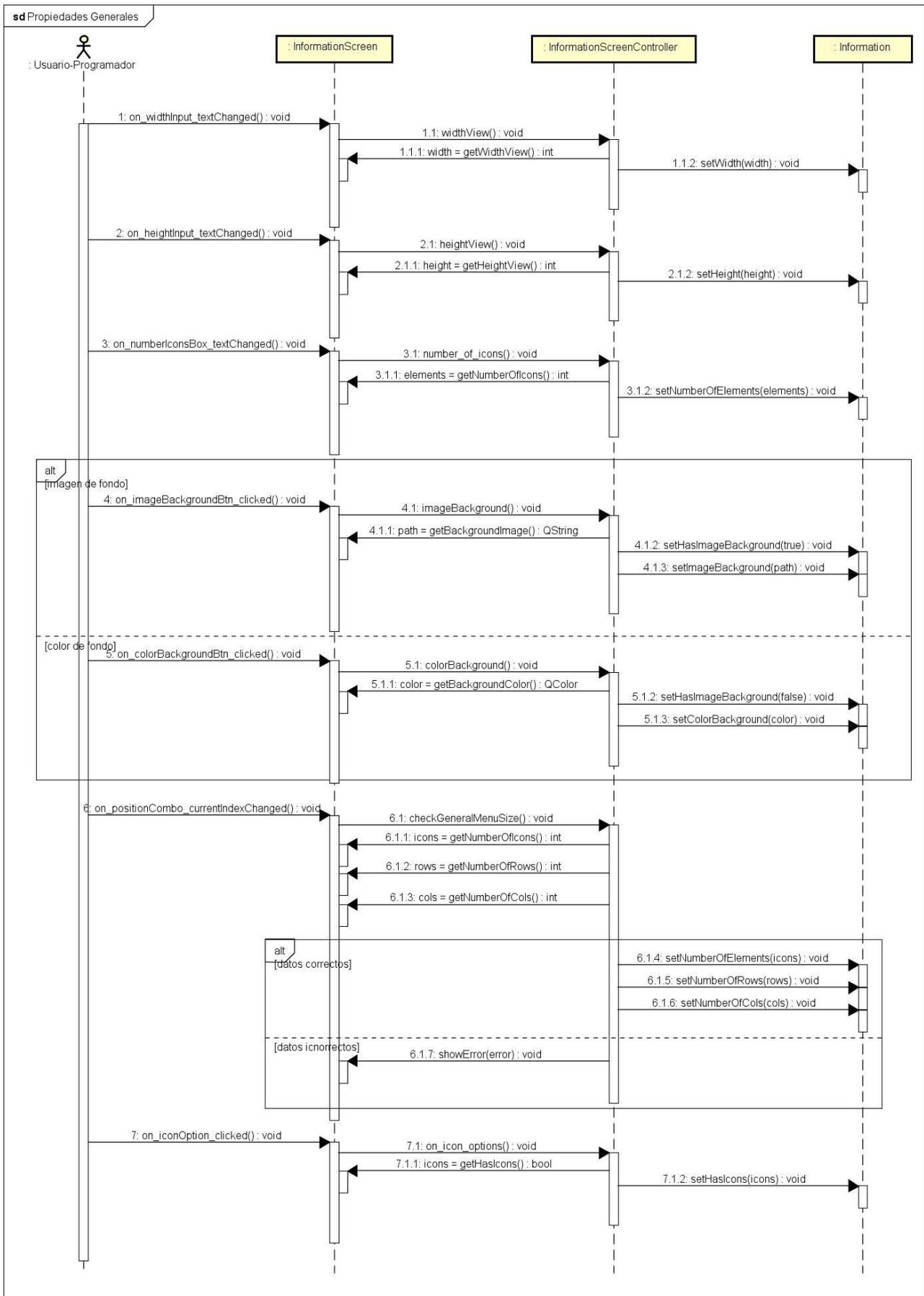


Figura 26. Diagrama de secuencia Propiedades Generales Información

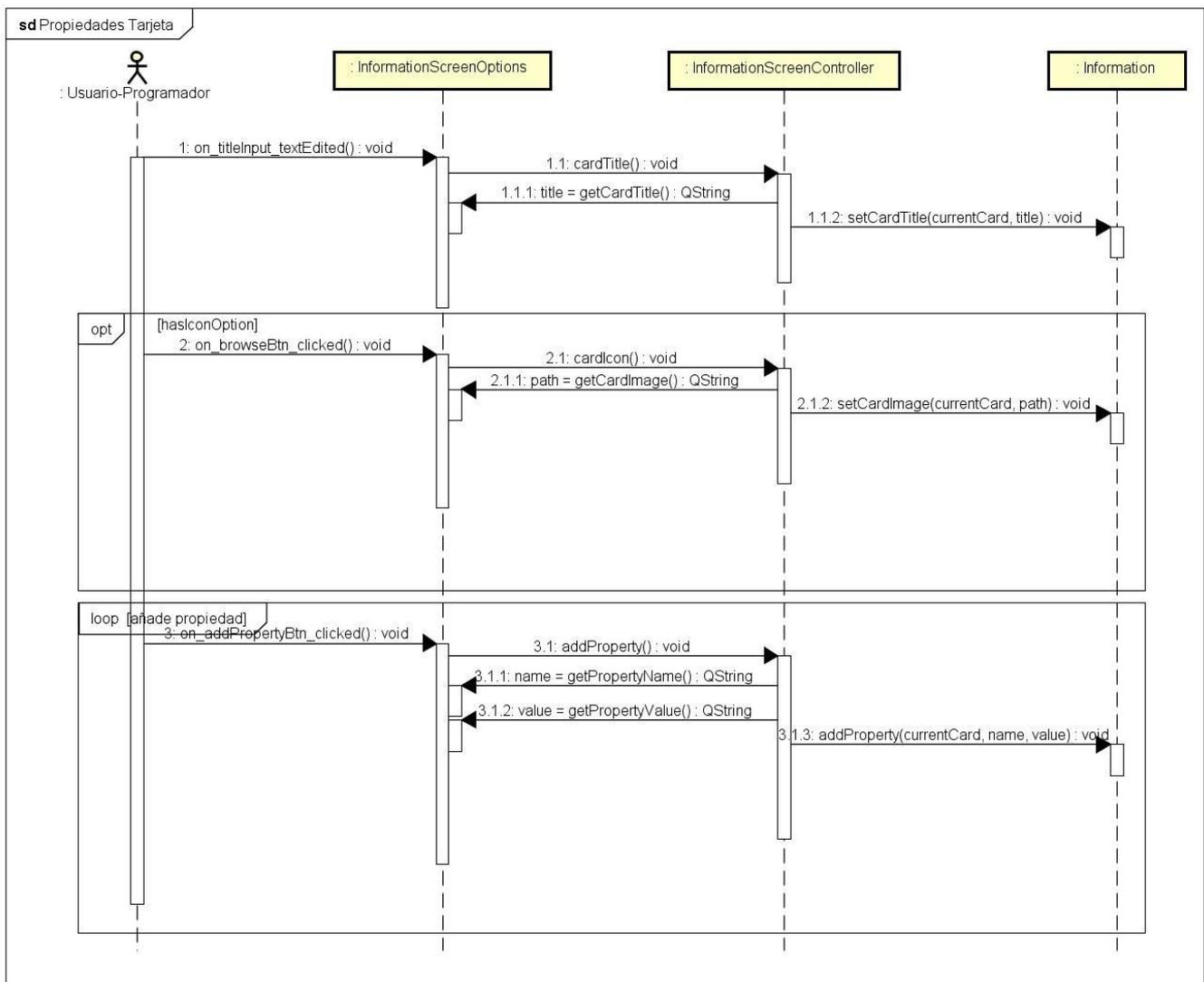


Figura 27. Diagrama de secuencia Propiedades Tarjeta Información

Creación interfaz tipo Tinta

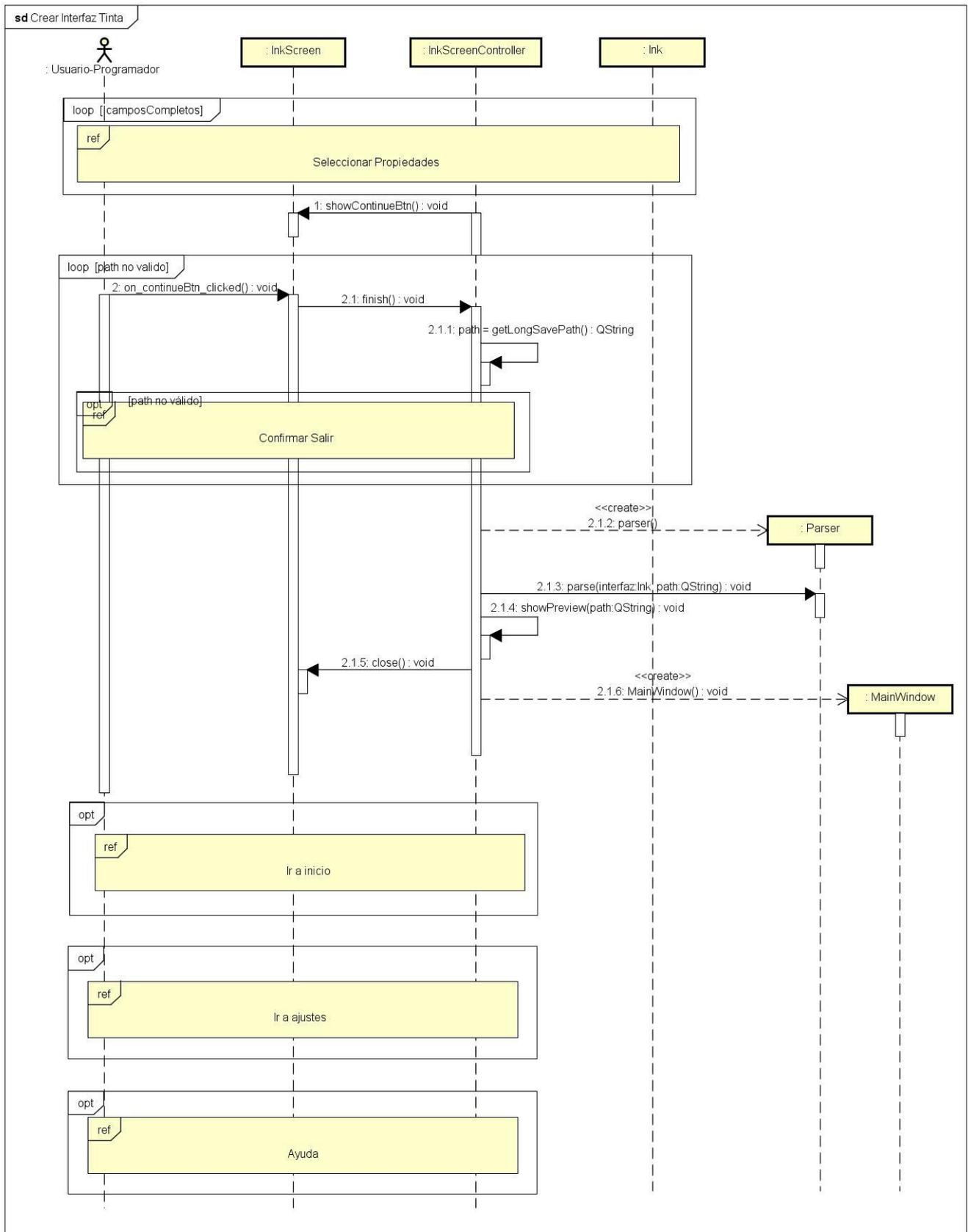


Figura 28. Diagrama de secuencia Creación General Tinta

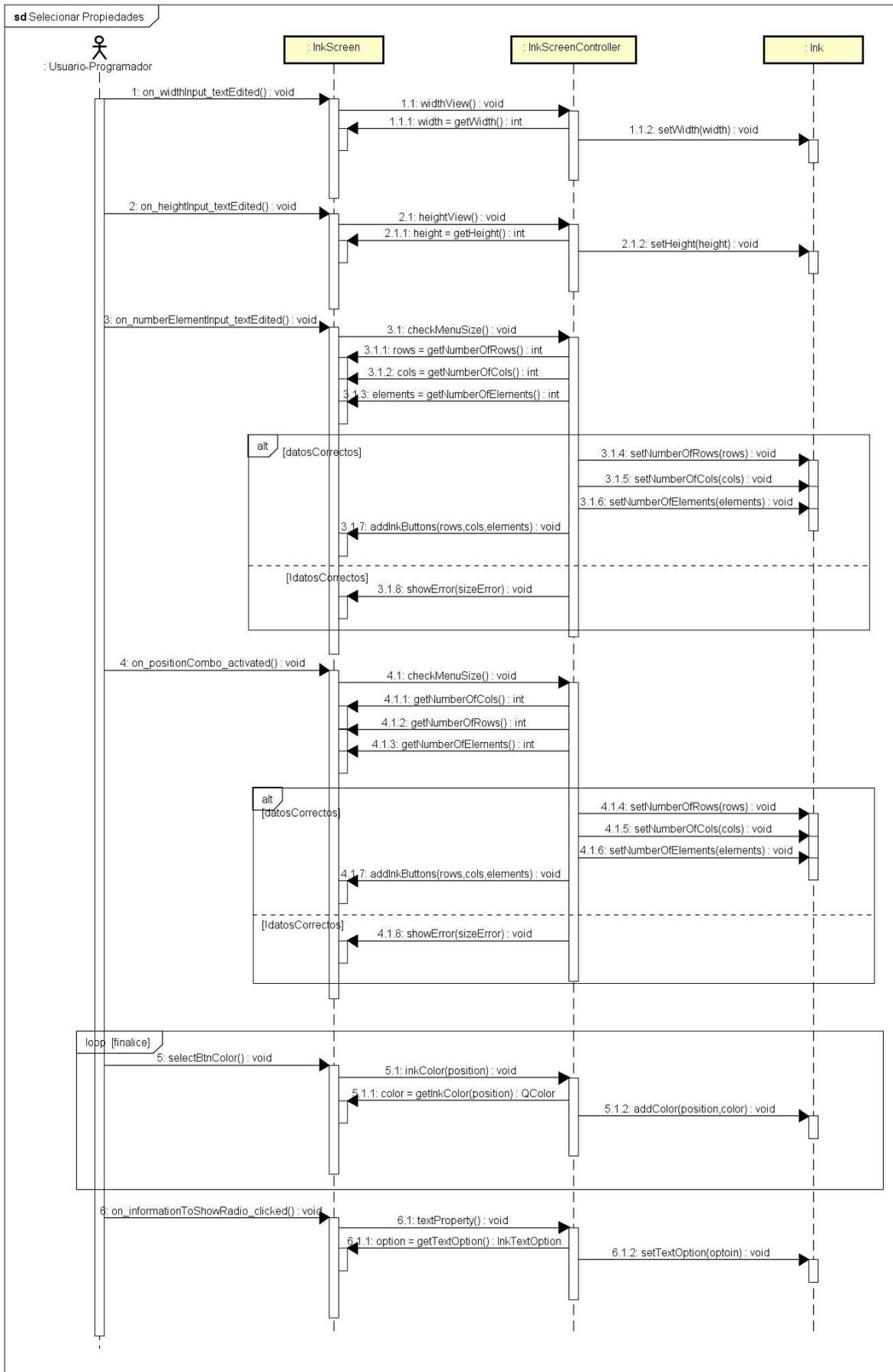


Figura 29. Diagrama de secuencia Seleccionar Propiedades Tinta

Edición

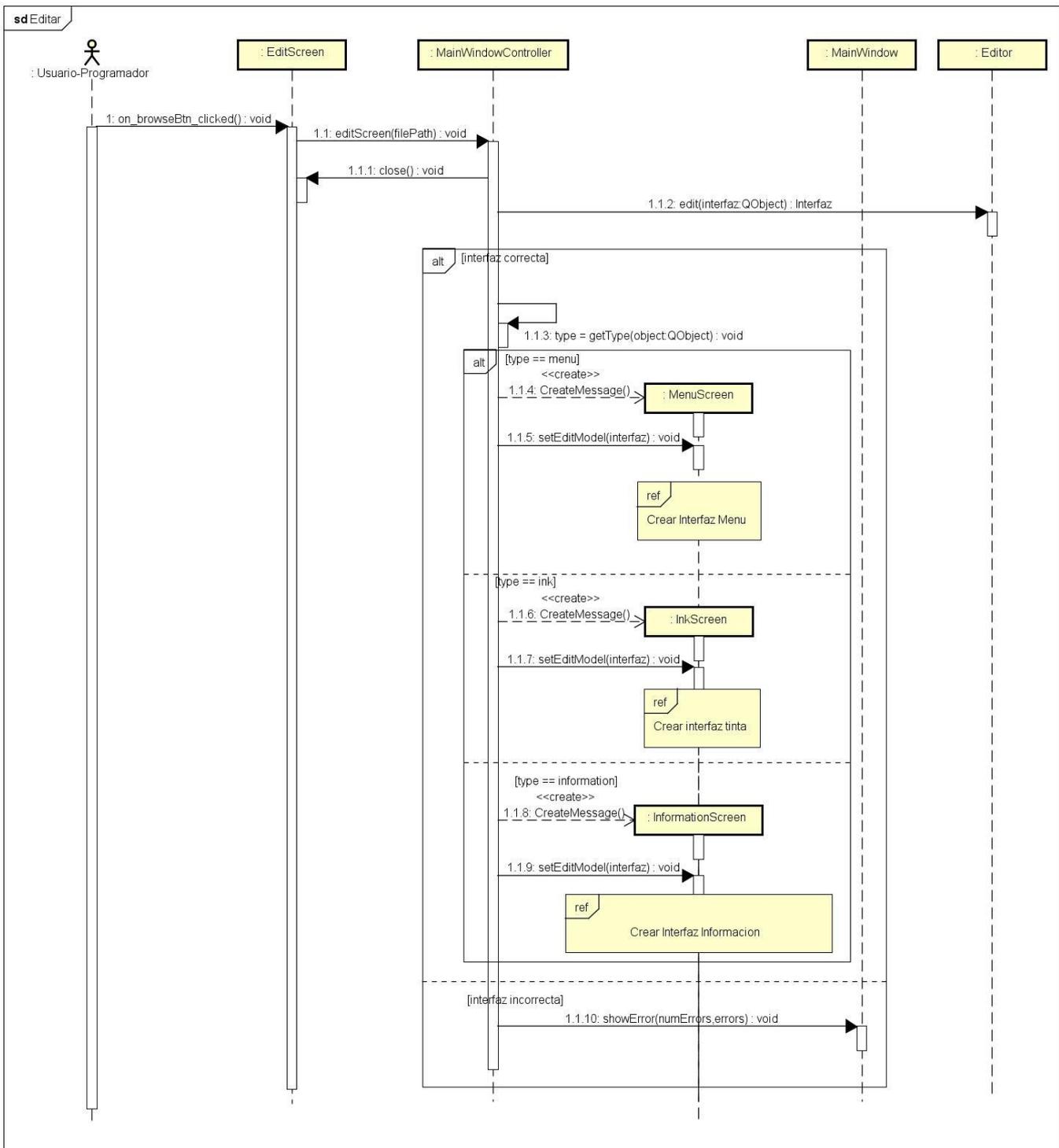


Figura 30. Diagrama de secuencia Editar

Capítulo 7. Implementación

En este capítulo se explica la forma de llevar a cabo la implementación de la aplicación.

Para llevar a cabo la implementación se ha utilizado un ordenador personal con Windows 10 y Ubuntu 16.04. Toda la aplicación se ha desarrollado utilizando la versión de Qt 5.8, el IDE Qt creator versión 4.2.0 y C++ como lenguaje de programación. El proyecto se divide en 3 paquetes siguiendo el patrón Modelo-Vista-Controlador, un paquete donde se encuentran las constantes necesarias para toda la aplicación y los servicios encargados de realizar las conversiones entre el modelo y la interfaz QML.

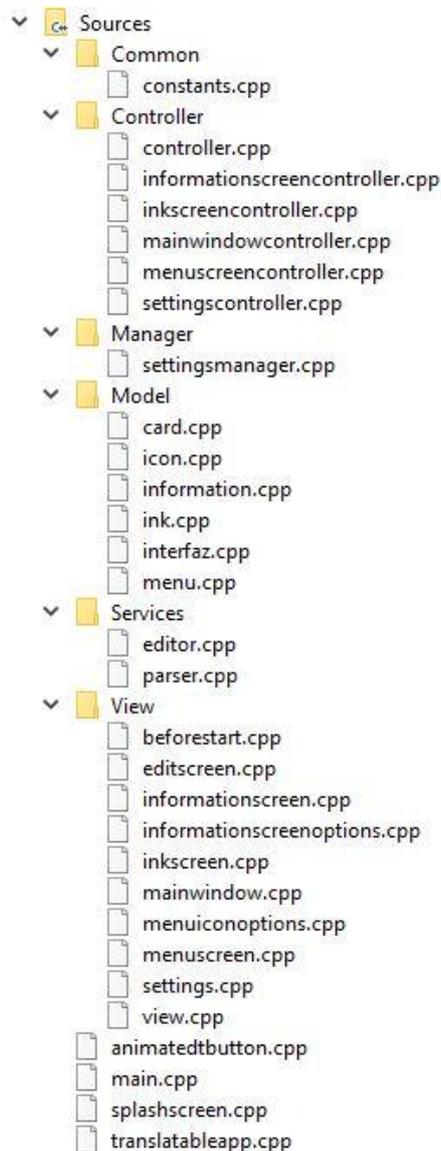


Figura 31 Organización del proyecto por paquetes

La organización de la implementación se ha realizado siguiendo la planificación establecida, es decir, en 6 etapas o hitos. En cada uno de ellos se especifica el trabajo realizado, los problemas encontrados y los problemas que se han solventado. En la mayoría de los casos los problemas de una etapa se solucionan la siguiente etapa.

Navegación básica

En esta primera etapa del proyecto se desarrolla la interfaz gráfica de toda la aplicación, así como la navegación básica de la misma. Para la realización de todas las ventanas se utiliza QMainWindow dado que proporciona, de forma nativa, las diferentes áreas en las que se divide las ventanas de la aplicación. [19]

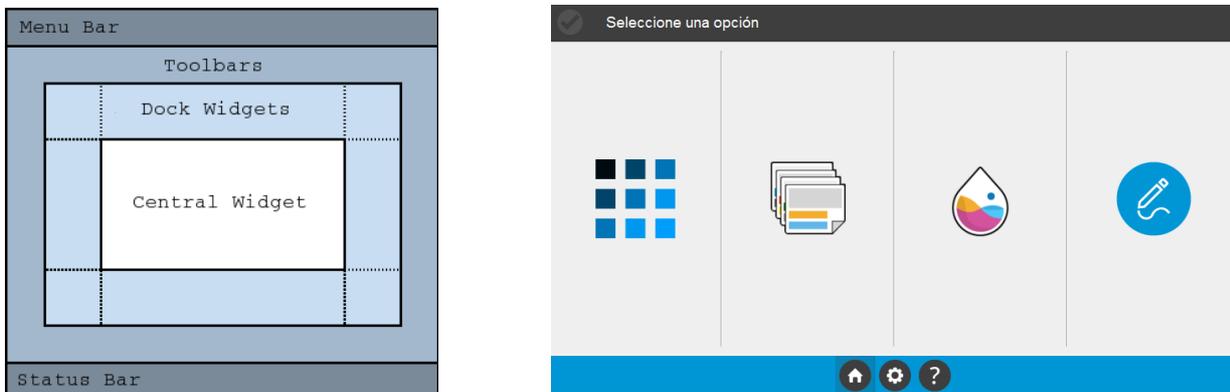


Figura 32 Ejemplo del esquema QMainWindow Qt

Estas áreas son 3:

- Barra superior de información donde se indica al usuario cual es el siguiente paso a realizar.
- Zona de trabajo o zona central donde el usuario realiza las acciones.
- Barra inferior de ajustes generales que permite la navegación general desde cualquier parte de la aplicación, como vuelta a inicio, ajustes o ir a la ayuda.

La zona central se encuentra implementada con diferentes componentes en función de lo que requiera la aplicación. Los componentes utilizados en todas las ventanas son los siguientes:

- QPushButton

Se trata del elemento más utilizado en toda la aplicación. Dado que se trata de un elemento principal, el cual activa o desactiva elementos o acciones dentro de la aplicación, se desarrolla un sistema mediante el cual el usuario dispone, en la mayoría de casos, de la opción de volver hacia atrás. De esta forma en el caso de que el usuario se equivoque mediante otro componente de este tipo puede volver a la situación previa. En los casos en los que la vuelta a atrás no es posible, antes de realizar la acción se pide una confirmación al usuario. [20]

- QLineEdit

Campo de edición de texto que se utiliza a la hora de crear campos en los que el usuario debe introducir valores personalizables. Por ejemplo: el número de elementos en una interfaz o el título de una propiedad. Con el fin de eliminar los errores por parte del usuario y la complejidad de las comprobaciones posteriores se restringe la capacidad de este elemento. Es decir, para campos numéricos, se establece que únicamente acepte caracteres numéricos restringidos a ciertos intervalos. De la misma forma, para campos de tipo texto, se permite introducir caracteres alfanuméricos, no permitiendo caracteres extraños o no conocidos. [21]

- QLayout

Componente imprescindible a la hora de gestionar la colocación de los elementos. Dentro de los diferentes layouts se encuentran los elementos de las interfaces. Siguiendo este esquema, cada interfaz se encuentra formada por diferentes tipos de layouts, ya sean verticales (QVBoxLayout) u horizontales (QHBoxLayout), los cuales se encuentran a su vez formados por todo tipo de elementos, incluidos otros layouts. [22] [23] [24]

- QComboBox

Se trata de un selector desplegable cuyo uso se centra en los casos en los que el usuario debe seleccionar una opción predeterminada en la que el número de opciones es superior a 2. De esta forma dispone de una visión general de todas las opciones permitiéndole elegir la que más le convenga en cada uno de los casos. [25]

- QRadioButton

En las situaciones en las que el número de opciones a seleccionar por el usuario son 2 se utiliza este elemento. Esta decisión se toma dado que se trata de botones de tipo exclusivo. Es decir, solamente uno de los elementos puede encontrarse pulsado a la vez, de esta forma, al tratarse de dos elementos, la selección se realiza de forma directa. En el caso del elemento QComboBox, para seleccionar una opción entre dos posibles, el usuario debería realizar una acción para poder desplegar las opciones y otra para seleccionar la que le convenga, aumentando así el número de pulsaciones necesarias. [26]

- QLabel

Campo de texto que muestra información al usuario. Todos los elementos a introducir por el usuario cuentan con un QLabel asociado de forma que siempre se sepa que campo se debe introducir en cada momento. Hay que destacar que dado que la aplicación dispone de diferentes idiomas se debe realizar una modificación en las propiedades de cada elemento de este tipo de forma que se permita su traducción. [27]

- QMessageBox

Este componente nativo de Qt permite crear mensajes a modo *pop-up* de diferentes tipos. Estos tipos varían entre avisos, errores, advertencias, etc. [28]

- QFileDialog

Selector de ficheros de Qt, por defecto utiliza el selector de ficheros del sistema operativo donde se encuentra. Este comportamiento por defecto atraerá problemas en las etapas finales de implementación que posteriormente se consiguen solucionar. [29]

- QColorDialog

Selector de colores de Qt, permite elegir al usuario sobre una paleta de colores idéntica a la utilizada en otros programas como los editores de imágenes. [30]

Un aspecto destacable de todos los elementos es la posibilidad de establecer políticas de crecimiento o tamaño. Estas propiedades determinan cómo se comporta cada uno de los elementos a

la hora de redimensionarse. En este tipo de configuraciones se invierte una gran cantidad de tiempo de desarrollo dado que toda la parte visual depende de su correcto funcionamiento.

Todas las ventanas de la aplicación siguen este esquema de forma que el usuario, una vez se encuentre familiarizado con el funcionamiento de una ventana, pueda navegar de forma transparente por toda la aplicación.

Creación de ventanas de tipo Menú

En una segunda etapa se comienza a realizar la parte *Core* de la aplicación, la creación de interfaces en formato QML, concretamente las interfaces de tipo Menú. Estas interfaces, al igual que cualquier cajón de aplicaciones de un dispositivo móvil, se encuentran formadas por iconos los cuales poseen, de forma opcional, un título. Partiendo de la implementación visual desarrollada en la etapa anterior, se elabora la parte correspondiente al modelo, así como la funcionalidad encargada de transformar dicho modelo en una interfaz QML.

La transformación entre el modelo, que representa a un menú, y la interfaz de tipo Menú se realiza partiendo de una plantilla. Dicha plantilla, se elabora, en un documento QML, de forma manual por el desarrollador. En este proceso se invierte una cantidad considerable de recursos dado que se trata de un punto clave para la aplicación. Conseguir elaborar una plantilla genérica que sirva para realizar los diferentes tipos de interfaces, puede ahorrar una cantidad considerable de horas de desarrollo. [31]

Durante este proceso se hace frente a diferentes problemas, entre los que destacan los dos siguientes:

- Edición de las interfaces generadas: toda la documentación encontrada hace referencia a la modificación de interfaces QML a partir de su correspondiente objeto C++. Es decir, primero se carga la interfaz en memoria a través de un objeto C++ y, posteriormente, se realiza la modificación del objeto. Al probar esta forma de implementación, se consiguen realizar las modificaciones necesarias sobre la plantilla. El problema radica en la imposibilidad de hacer persistentes dichos cambios. [32]
- Selección de imágenes: para crear las interfaces de tipo menú es necesario poder introducir imágenes. Estas imágenes se usarán como iconos del menú y, en el caso de que el usuario lo seleccione, como fondo del menú. El problema encontrado en esta etapa se encuentra relacionado con la inclusión de la imagen en la interfaz QML. La documentación encontrada ilustra como incluir imágenes ya cargadas en los recursos de la aplicación, no se conoce como se puede mostrar una imagen cualquiera del sistema. Como solución, se establece que, en etapas posteriores, se va a implementar un selector de imágenes que muestre todas las imágenes precargadas en el sistema. Esta solución posee un hándicap, el usuario solamente podrá añadir iconos que se encuentren cargados en el sistema. Para paliar esta situación, y tras realizar investigación sobre los recursos en Qt, se halla la forma de poder permitir a la aplicación usar un tipo concreto de ficheros de recursos externos. Esta funcionalidad se implementará en etapas posteriores. [33]

Creación de ventanas de tipo Información

Durante el desarrollo de esta etapa se realiza la implementación correspondiente a la creación de interfaces de tipo información. Para ello, se sigue el mismo esquema utilizado en la etapa anterior. Este desarrollo se realiza en el tiempo establecido dado que ya se contaba con la experiencia adquirida, así como de la plantilla a partir de la cual se generan las interfaces. Las interfaces de tipo información se encuentran formadas por tarjetas. Cada una de estas tarjetas dispone de elementos de texto y, opcionalmente, de una imagen o icono.

A la hora de la implementación de los elementos QML se realizan pruebas de posibles mejoras. La más significativa es la posibilidad de introducir una cabecera o un pie en la interfaz. Las pruebas se realizan con el fin de poder añadir dicho elemento a las interfaces generadas de forma que la navegación, una vez que la interfaz se encuentre en uso por un tercero, sea directa.

Por otro lado, con el fin de mejorar la usabilidad de la aplicación se hace uso de la ayuda contextual. Para ello se observa el uso otras aplicaciones de escritorio y se decide reproducir su funcionamiento. Cuando un usuario mantiene el cursor encima de un elemento se le muestra una información asociada a dicho elemento. Esta información indica al usuario que acción realiza el elemento. [34]

Como en casos anteriores en los que se introducen textos en la aplicación, se deben realizar las modificaciones necesarias en dichos elementos con el fin de que permitan su traducción.

Además, también se encuentra un problema crítico en la creación de interfaces QML. El problema radica en el tamaño y la redimensión de los diferentes elementos dentro de las interfaces de tipo menú e información. El elemento gráfico o icono de las interfaces se redimensiona de forma que ocupe un porcentaje de la interfaz, dejando el resto para los elementos de texto. Esta implementación produce que en las ocasiones en las que el usuario introduce textos de longitud más larga, el tamaño del icono sea desproporcionado con respecto al texto. Tras realizar una investigación se encuentra la posibilidad de realizar la redimensión de los elementos dependiendo del tamaño del texto a introducir en la interfaz. Para ello se debe obtener el tamaño del texto a partir de QFontMetrics. Esta implementación se retrasa para etapas posteriores. [35]

Creación de ventanas de tipo Nivel de Tinta

Dentro de esta etapa se realiza el desarrollo correspondiente al último tipo de interfaces, las interfaces de tipo tinta. Se trata del tipo más simple de interfaz dado que los elementos utilizados para formar parte de la interfaz se encuentran entre los tipos genéricos de QML.

Uno de los problemas encontrados durante esta etapa es el relacionado con la selección de los colores de los niveles de tinta. Dado que el número de niveles de tinta es un parámetro a introducir por el usuario, la interfaz se debe generar de forma dinámica. La dificultad se encuentra a la hora de establecer unos controles que faciliten la usabilidad de la aplicación. Finalmente se decide implementar una disposición de forma que cada selector de color se sitúa en la misma posición que ocupará el nivel de tinta correspondiente dentro de la interfaz.

También se desarrollan implementaciones pendientes de etapas anteriores como el selector de iconos que se introdujo en la etapa de creación de ventanas de tipo menú. Como ya se dijo anteriormente, este selector se realiza con el fin de permitir al usuario elegir las imágenes precargadas en el sistema. Durante el desarrollo de este selector se consigue una forma de establecer cualquier tipo de imagen, se encuentre cargada en la aplicación o no.

Como parte de las implementaciones menores necesarias para mejorar la usabilidad de la aplicación se realizan los ajustes. Estos se encuentran diseñados siguiendo los esquemas seguidos en otras aplicaciones. Dentro de los ajustes se permite al usuario la modificación del idioma o establecer un directorio por defecto. Para ello se utiliza QSettings, una clase que provee Qt de forma propia para mantener valores de la aplicación entre sesiones de la misma. Para ello hace uso de un fichero, que se genera de forma automática a través de un objeto QSettings. Dentro de estos ajustes se pueden guardar una gran variedad de tipos de objetos. Una vez se recuperan, Qt nos provee de un el tipo QVariant. que actúa de contenedor genérico a través del cual poder obtener todos los valores almacenados. [36] [37]

Corrección de errores I

Durante esta etapa los recursos se centran en detectar y arreglar los errores y fallos en la creación de interfaces de tipo menú y tipo tinta. Para ello se destinan todos los recursos a la realización sistemática de pruebas de diferentes tipos.

Se realizan también otras implementaciones menores como son la posibilidad de inserción de cualquier tipo de imagen dentro de las interfaces, así como la ayuda dentro de la aplicación.

Para conseguir mostrar cualquier tipo de imagen dentro de las interfaces se realiza un trabajo de investigación y de pruebas. Se encuentra la forma de proveer de imágenes a las interfaces QML a partir de la carga de la imagen dentro de los recursos. Al tratarse de una documentación no oficial se realizan diferentes pruebas con el fin de determinar el rendimiento de la nueva funcionalidad. Realizando dichas pruebas se determina que cualquier interfaz en QML puede mostrar imágenes a través de su enlace a la página de internet. A partir de este descubrimiento se comienzan a realizar pruebas de acceso al sistema de ficheros desde el navegador web de forma que la ruta de la imagen local fuese la proporcionada por el navegador. De esta forma se consigue que las interfaces QML muestren las imágenes locales. [38]

En el caso de la ayuda se realiza una primera implementación fallida partiendo de elementos nativos de Qt. Posteriormente se realiza una segunda implementación en la que se hace uso del navegador por defecto del sistema.

La ayuda de la aplicación se basa en diferentes páginas web, una para cada ventana de la aplicación. En la primera aproximación se intenta mostrar dichas páginas haciendo uso de QTextBrowser y QTextEdit, dos componentes utilizados para mostrar y editar elementos web dentro de Qt. Durante el desarrollo nos encontramos con el problema de que estos componentes no soportan toda la funcionalidad de HTML o CSS. Se encuentra la posibilidad de añadir módulos propios de Qt de forma que soporten toda la funcionalidad de las tecnologías web, el inconveniente el tamaño de dichos paquetes. Por ello se decide descansar la visualización de estos elementos sobre el navegador web

del sistema. Haciendo uso de QDesktopServices, se realiza una llamada al sistema operativo donde se encuentra la aplicación, y se obtiene la visualización de la ayuda. [39] [40] [41]

Corrección de errores II

Aunque la planificación establece que durante esta etapa de desarrolla la edición de los tres tipos de interfaces, durante esta etapa únicamente se realiza la edición de interfaces de tipo tinta dado que es el tipo de interfaz que menos elementos posee. El retraso viene condicionado por una implementación anterior que obliga a modificar las plantillas a partir de las cuales se generan las interfaces.

La edición de las interfaces se lleva a cabo a través del sistema de meta modelo que posee Qt. Gracias a este sistema se puede acceder a las propiedades y sus valores de los objetos QML. En este punto es donde cobra importancia el hecho de que todos los elementos QML utilizados sean elementos generales y no personalizados, dado que de ser así el desarrollo se hubiese prolongado dada la necesidad de declarar dentro del meta modelo QML las nuevas propiedades. Mediante este sistema accedemos a las propiedades de las interfaces y creamos el modelo que representa dicha interfaz dentro del sistema de forma que el usuario pueda editar la interfaz. [42] [43] [44]

Durante esta etapa se realiza otra modificación fundamental. Esta viene asociada con la plantilla QML a partir de la cual se generan todos los tipos de interfaces de la aplicación. Esta modificación viene condicionada del error de redimensión encontrado en la etapa de creación de ventanas de tipo información. Siguiendo la idea inicial, la aplicación debería medir la longitud del texto a introducir en cada componente de la interfaz haciendo uso de QFontMetrics, y a partir de ahí redimensionar los elementos de forma equitativa. En el caso de haber realizado dicha implementación se deberían realizar los cálculos cada vez que se crease y editase una interfaz. Con el fin de facilitar la implementación de la edición se implementa dicha redimensión utilizando la potencia de QML. Gracias a las modificaciones realizadas sobre las diferentes plantillas las interfaces se redimensionan dependiendo del tamaño de los elementos sin necesidad de realizar ningún tipo de cálculo o variación.

Finalización

En este último periodo se realiza la edición de las interfaces de tipo menú y de tipo información. Para ello se utiliza la misma técnica que se utilizó en la etapa anterior para las interfaces de tipo tinta. A partir del meta modelo accedemos a las propiedades de la interfaz y permitimos al usuario su modificación. Como mejora, se introduce una funcionalidad bastante útil a la hora de editar las interfaces. Esta mejora se centra en la retroalimentación al usuario de los errores encontrados en el caso de no poder editar la interfaz. De esta forma, si el usuario edita la interfaz de forma externa y los cambios no son aceptados por el asistente se comunica de forma clara la ubicación de los errores.

Hay que destacar que durante esta etapa se encuentra una situación excepcional en la que la aplicación deja de funcionar de forma correcta. Tras el análisis de la aplicación se consigue obtener la causa del problema, el cual es un *Bug* en la versión de Qt 5.8 y Windows 10 relacionado con el selector de ficheros nativo de Qt. Este selector descansa, por defecto, sobre el selector de ficheros del sistema operativo en el que se encuentra la aplicación. Como solución, se decide que el selector a utilizar sea el nativo de Qt, de forma que no utilice el propio del sistema operativo. [45]

Capítulo 8. Pruebas

8.1. Pruebas de caja negra

Home	
ID	PR – 001
Descripción	El usuario solicita volver a la pantalla principal
Resultado Esperado	El sistema muestra un mensaje de confirmación y vuelve a la ventana principal del sistema sin generar la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 19. Prueba 001 - Home

Ayuda	
ID	PR – 002
Descripción	El usuario solicita la ayuda en cualquier parte del sistema
Resultado Esperado	El sistema abre el navegador por defecto del sistema y muestra la ayuda correspondiente a la situación del usuario en el sistema
Estado	Finalizado
Funcionamiento	Correcto

Tabla 20. Prueba 002 - Ayuda

Ajustes	
ID	PR – 003
Descripción	El usuario solicita abrir los ajustes
Resultado Esperado	El sistema muestra una ventana de confirmación y muestra la ventana de ajustes
Estado	Finalizado
Funcionamiento	Correcto

Tabla 21. Prueba 003 - Ajustes

Restaurar Ajustes	
ID	PR – 004
Descripción	El usuario restablece los valores por defecto
Resultado Esperado	El sistema borra los valores de los ajustes y muestra un mensaje de éxito
Estado	Finalizado
Funcionamiento	Correcto

Tabla 22. Prueba 004 – Restaurar Ajustes

Acerca de	
ID	PR – 005
Descripción	El usuario solicita información “acerca de”
Resultado Esperado	El sistema muestra una ventana con información sobre la aplicación
Estado	Finalizado
Funcionamiento	Correcto

Tabla 23. Prueba 005 – Acerca de

Interfaz de tipo menú	
ID	PR – 006
Descripción	El usuario realiza el proceso de creación de una interfaz de tipo menú
Resultado Esperado	El sistema genera una interfaz de tipo menú almacenándola en un fichero QML
Estado	Finalizado
Funcionamiento	Correcto

Tabla 24. Prueba 006 – Creación interfaz menú

Cancelar creación menú	
ID	PR – 007
Descripción	El usuario cancela la creación del menú durante el proceso
Resultado Esperado	El sistema vuelve a la ventana principal del sistema sin generar la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 25. Prueba 007 – Cancelar creación menú

Navegación menú	
ID	PR – 008
Descripción	El usuario no introduce todos los campos durante el proceso de creación de una interfaz de tipo menú
Resultado Esperado	El sistema no le permite avanzar en el proceso de creación
Estado	Finalizado
Funcionamiento	Correcto

Tabla 26. Prueba 008 - Navegación menú

Creación interfaz información	
ID	PR – 009
Descripción	El usuario realiza el proceso de creación de una interfaz de tipo información
Resultado Esperado	El sistema genera una interfaz de tipo información almacenándola en un fichero QML
Estado	Finalizado
Funcionamiento	Correcto

Tabla 27. Prueba 009 – Creación interfaz información

Cancelar creación información	
ID	PR – 010
Descripción	El usuario cancela la creación de una interfaz de tipo información durante el proceso
Resultado Esperado	El sistema vuelve a la ventana principal del sistema sin generar la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 28. Prueba 010 - Cancelar creación información

Navegación información	
ID	PR – 011
Descripción	El usuario no introduce todos los campos durante el proceso de creación de una interfaz de tipo información
Resultado Esperado	El sistema no le permite avanzar en el proceso de creación
Estado	Finalizado
Funcionamiento	Correcto

Tabla 29. Prueba 011 – Navegación información

Creación interfaz tinta	
ID	PR – 012
Descripción	El usuario realiza el proceso de creación de una interfaz de tipo tinta
Resultado Esperado	El sistema genera una interfaz de tipo tinta almacenándola en un fichero QML
Estado	Finalizado
Funcionamiento	Correcto

Tabla 30. Prueba 012 – Creación interfaz tinta

Cancelar creación tinta	
ID	PR – 013
Descripción	El usuario cancela la creación de la interfaz de tipo tinta durante el proceso
Resultado Esperado	El sistema vuelve a la ventana principal del sistema sin generar la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 31. Prueba 013 – Cancelar creación tinta

Navegación interfaz tinta	
ID	PR – 014
Descripción	El usuario no introduce todos los campos durante el proceso de creación de una interfaz de tipo tinta
Resultado Esperado	El sistema no le permite avanzar en el proceso de creación
Estado	Finalizado
Funcionamiento	Correcto

Tabla 32. Prueba 014 – Navegación interfaz tinta

Selección de imagen de fondo	
ID	PR – 015
Descripción	El usuario selecciona una imagen como fondo para una interfaz
Resultado Esperado	El sistema vuelve a la ventana de creación previa indicando que la imagen de fondo se ha seleccionado correctamente
Estado	Finalizado
Funcionamiento	Correcto

Tabla 33. Prueba 015 – Selección imagen de fondo

Cancelar selección de imagen de fondo	
ID	PR – 016
Descripción	El usuario cancela la selección de la imagen de fondo de una interfaz
Resultado Esperado	El sistema vuelve al punto de creación previo a la selección mostrando un mensaje de que no se ha establecido un fondo para la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 34. Prueba 016 – Cancelar selección imagen de fondo

Selección de color de fondo	
ID	PR – 017
Descripción	El usuario selecciona un color de fondo para una interfaz
Resultado Esperado	El sistema vuelve a la ventana de creación previa indicando que el color de fondo se ha establecido correctamente
Estado	Finalizado
Funcionamiento	Correcto

Tabla 35. Prueba 017 – Selección color de fondo.

Cancelar selección de color de fondo	
ID	PR – 018
Descripción	El usuario cancela la selección de un color de fondo para una interfaz
Resultado Esperado	El sistema vuelve al punto de creación previo a la selección mostrando un mensaje de que no se ha establecido un fondo para la interfaz
Estado	Finalizado
Funcionamiento	Correcto

Tabla 36. Prueba 018 – Cancelar selección de color de fondo

Edición	
ID	PR – 019
Descripción	El usuario solicita editar una interfaz válida
Resultado Esperado	El sistema muestra las propiedades de la interfaz dependiendo de su tipo
Estado	Finalizado
Funcionamiento	Correcto

Tabla 37. Prueba 019 – Edición interfaz

Error edición	
ID	PR – 020
Descripción	El usuario solicita editar una interfaz no válida
Resultado Esperado	El sistema muestra un mensaje de error indicando los detalles del mismo
Estado	Finalizado
Funcionamiento	Correcto

Tabla 38. Prueba 020 - Error edición

Cancelar edición	
ID	PR – 021
Descripción	El usuario solicita editar una interfaz y cancela la selección del fichero
Resultado Esperado	El sistema vuelve a la ventana principal
Estado	Finalizado
Funcionamiento	Correcto

Tabla 39. Prueba 021 – Cancelar edición

Capítulo 9. Tecnología Empleada

9.1. C++

C++ es un lenguaje de programación de propósito general, orientado a objetos e imperativo. Diseñado a mediados de la década de 1980 por Bjarne Stroustrup con la intención de extender la funcionalidad del lenguaje de programación C de forma que permitiese la manipulación de objetos.



Figura 33 Icono lenguaje c++

Inicialmente destaca por su eficiencia y rendimiento en los sistemas y equipos empujados, donde los recursos son escasos. Posteriormente, se utiliza su potencia en muchos otros contextos incluyendo las aplicaciones de escritorio. Posee un estándar, el ISO C++, a partir del cual se han desarrollado la mayoría de compiladores modernos. [46] [47]

9.4. Qt Creator

Qt Creator es un entorno de desarrollo integrado o IDE multiplataforma, programado en C++, JavaScript y QML, que se ajusta a las necesidades de los desarrolladores Qt. Forma parte del proyecto Qt Project.



Figura 34 Icono Qt Creator

Se centra en el aprendizaje del usuario y facilita el desarrollo mediante diferentes funcionalidades: [48] [49]

- Un editor de código con soporte para C++, QML y ECMAScript con todas las características propias de otros editores como el autocompletado, resaltado de sintaxis, soporte para el refactoring de código, etc.
- Depurador visual para C++ permitiendo establecer puntos de ruptura, ejecución línea por línea, visionado de variables, etc.
- Diseñador de interfaces gráficas de usuario integrado.

Capítulo 10. Manuales

En este manual se describe el funcionamiento de las diferentes opciones o características de la aplicación. Para ello se detalla su la forma de trabajar, paso a paso, completando la ayuda con capturas reales de la aplicación.

10.1. Manual de instalación

Para instalar la aplicación se hace uso de un instalador. Dicho instalador se trata de un fichero ejecutable que nos facilita y automatiza las configuraciones iniciales de la aplicación. Una vez comencemos con la instalación se muestra un mensaje de bienvenida. Posteriormente se

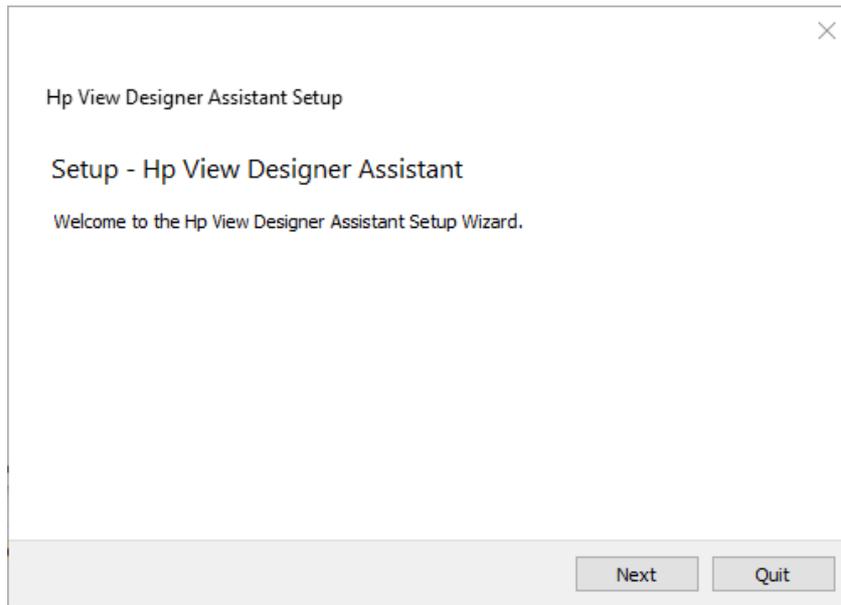


Figura 35 Mensaje bienvenida instalador

indica al usuario que debe seleccionar un directorio donde realizar la instalación.

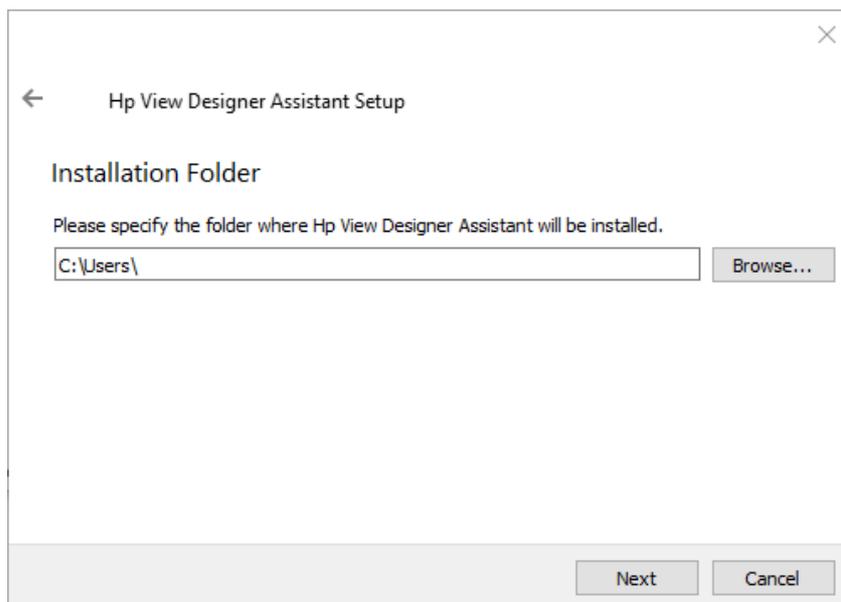


Figura 36 Selector directorio instalador

Se solicita al usuario que aplicación de las que componen el sistema desea instalar. Al tratarse de una única aplicación no disponemos de otras opciones. Tras esto, el usuario debe introducir el nombre del menú de inicio en el que quiere que aparezca el asistente.

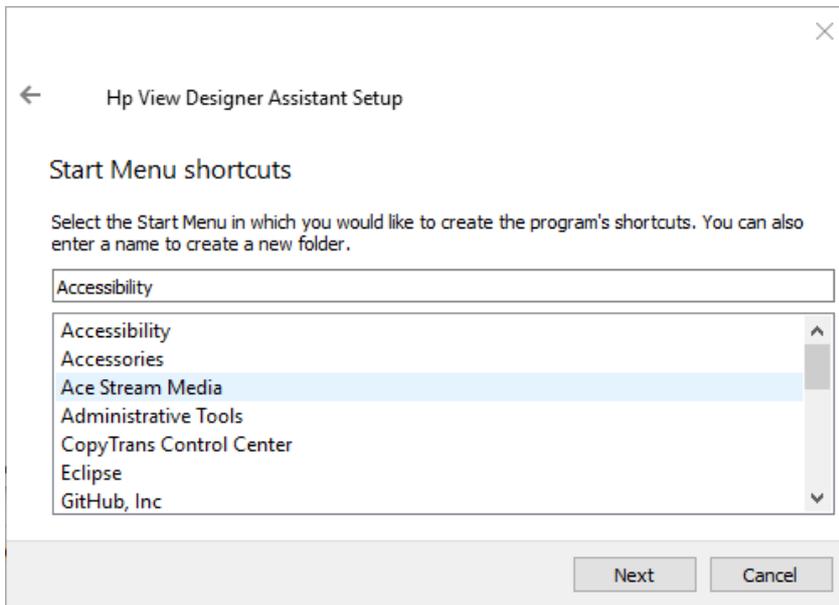


Figura 37 Selección menú inicio instalador

En el siguiente paso el asistente pide al usuario confirmar la instalación. En el caso de que el usuario acepte se realiza la instalación mostrando un progreso de la misma

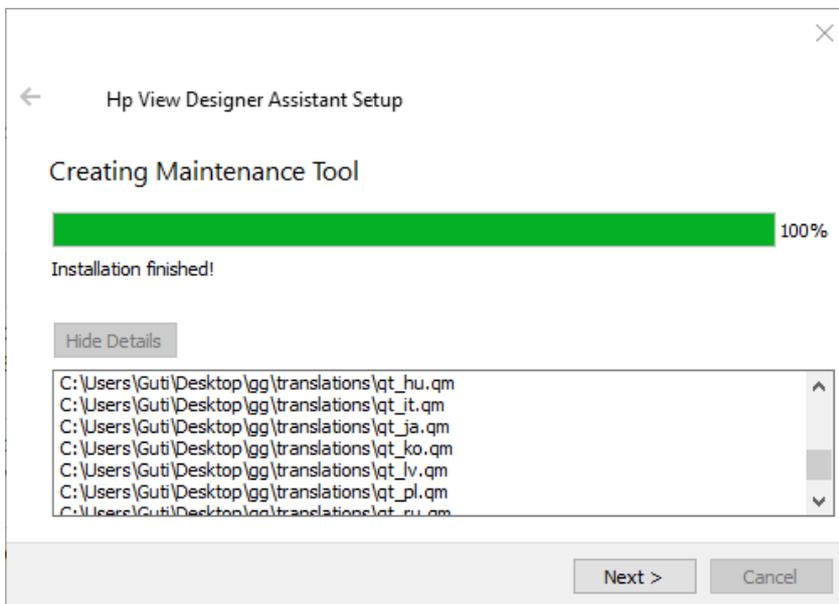


Figura 38 Progreso instalador

Por último, el asistente muestra un mensaje indicando que la instalación ha finalizado.

10.2. Manual de usuario

Una vez iniciada la aplicación se observa la pantalla de bienvenida, la cual es una animación en la que se muestra el título del proyecto. En un segundo plano y de forma simultánea, la aplicación realiza la creación de los recursos necesarios para su funcionamiento. Tras finalizar, se oculta la pantalla de bienvenida y en su lugar aparece la ventana principal de la aplicación, así como una pequeña guía explicando de forma general el funcionamiento de la aplicación.



Figura 39 Animación inicio



Figura 40 Ventana Antes de Comenzar

Esta ventana cuenta la información básica sobre la aplicación de forma que el usuario se haga una idea general de su funcionamiento y capacidad. En la última de las ventanas que forman este mensaje de bienvenida se encuentra un botón para comenzar, así como la posibilidad de marcar a la aplicación de que no vuelva a mostrar el mensaje de inicio. La aplicación no podrá utilizarse hasta que el usuario pulse el botón comenzar en la última ventana o cierre la ventana pulsando el botón general de cerrar situado en la esquina superior derecha.

10.3. Ventana principal

La ventana principal es el punto de partida para todas las funcionalidades de la aplicación. En esta ventana encontramos dos secciones claramente diferenciadas:

- Opciones principales
- Opciones secundarias

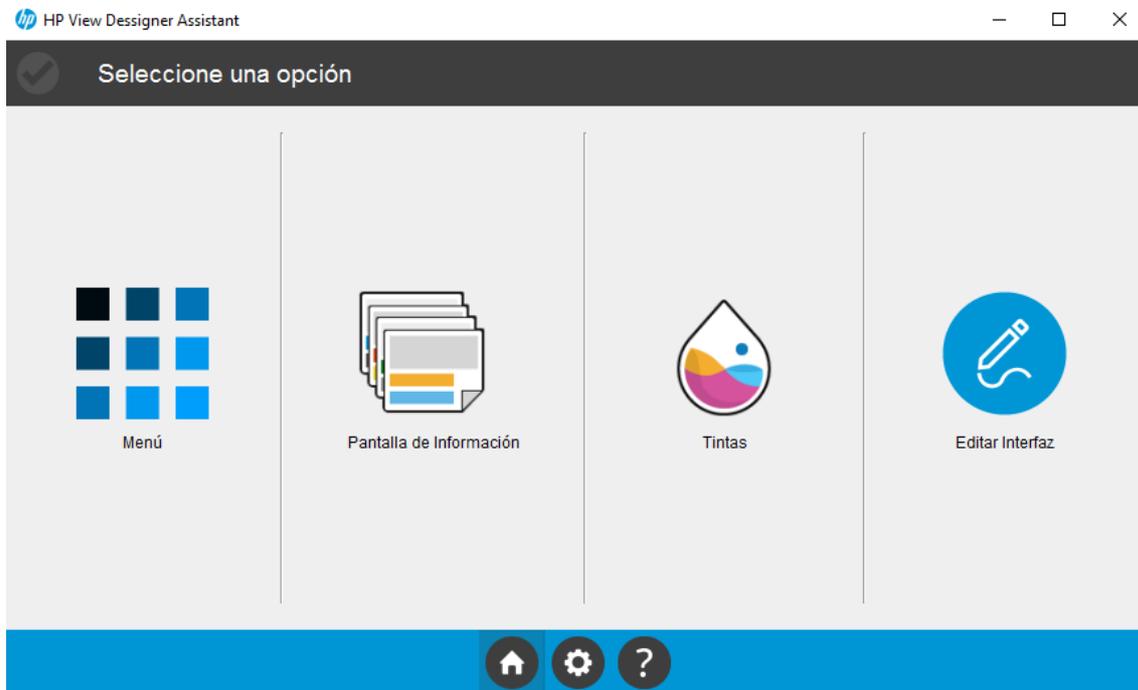


Figura 41 Ventana principal de la aplicación

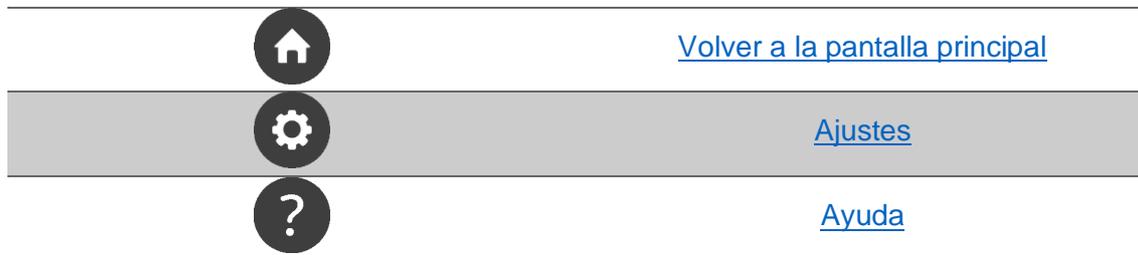
Opciones principales

Las opciones principales se centran en las dos funcionalidades principales de la aplicación: creación e edición de interfaces.



Opciones Secundarias

En este segundo caso se encuentran funcionalidades básicas, aunque comunes a casi todas las aplicaciones actuales. Estas tres opciones se encuentran, de forma permanente, en la parte inferior de todas las ventanas. De esta forma la navegación básica está garantizada en cualquier punto de la aplicación. Estas opciones son:



10.4. Creación

Es la funcionalidad básica de la aplicación, la creación de interfaces en formato QML. El usuario dispone de tres opciones de creación, las cuales son:

- Menús
- Información
- Tinta

Cada uno de los tres tipos posee un proceso independiente de creación, aunque poseen muchas similitudes entre ellos. En los siguientes apartados se explica cómo se realiza el proceso de creación de cada uno de los tipos.

Una vez finaliza cualquiera de los procesos de creación la aplicación muestra una vista previa de cómo ha quedado la interfaz generada. De esta forma, el usuario puede determinar si cumple los requisitos esperados y, de no ser así, puede proceder a editarla desde la propia aplicación.

10.4.1. Menú

Esta opción permite al usuario crear interfaces de tipo menú. Para hacer uso de esta funcionalidad el usuario debe pulsar el icono de Menú que se encuentra en la ventana principal de la aplicación.

Para poder realizar este proceso se deben establecer una serie de parámetros, estos se dividen en dos tipos:

- Parámetros generales de la interfaz: estos parámetros son comunes a la interfaz, como el tamaño, el fondo, el número de iconos del menú, etc.

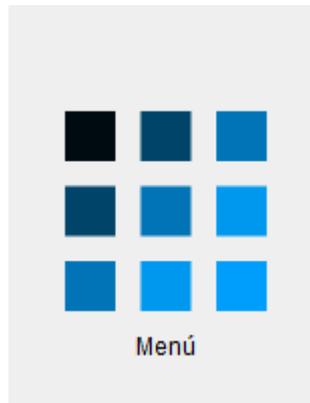


Figura 42 Icono creación menú

- Parámetros de cada icono: estos son personalizables para cada icono dentro de la interfaz, como el icono, el nombre del icono, etc.

Una vez se encuentren rellenos todos los datos de cada ventana el usuario podrá continuar con el siguiente paso en el proceso.

Parámetros generales del menú

En esta ventana podemos diferenciar seis parámetros diferentes para personalizar la interfaz.

1. Tamaño y fondo

En este apartado agrupamos los dos primeros parámetros dado que ambos afectan a la totalidad de la interfaz, no a elementos dentro de ella.



Figura 43 Propiedades generales del menú

Las dimensiones de la nueva interfaz vienen definidas por su alto y su ancho. Ambos valores se introducen en píxeles.

Para elegir el fondo de la interfaz disponemos de la opción de elegir una imagen o un color. En la primera opción una vez hagamos clic en el icono se nos abre un dialogo para poder elegir el fichero a utilizar entre todos los que se encuentren en el sistema. Si se hace clic en



Figura 44 Botones selección de fondo Menú

la opción de color se nos abre un dialogo en el que se nos muestra una paleta en la que podremos seleccionar el color deseado.

Una vez completemos una de las dos opciones y si todo ha ido bien, se nos mostrará un mensaje indicando que el fondo se ha establecido correctamente. En el caso de que se encuentre algún error, se mostrará un mensaje al usuario indicándole el problema.

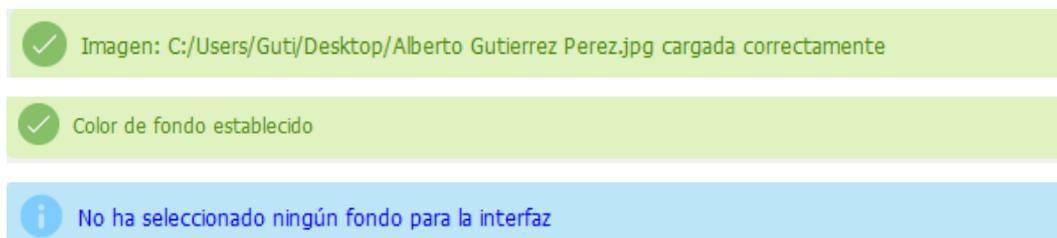


Figura 45 Mensajes de información menú

Para modificar el fondo establecido únicamente es necesario hacer clic en una de las dos opciones y seleccionar el nuevo fondo.

2. Número de iconos

Este parámetro establece el número de iconos con los que va a contar la interfaz.

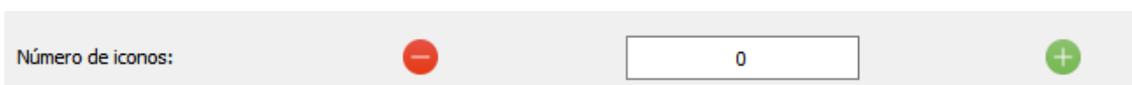


Figura 46 Zona edición número de iconos del menú

El parámetro puede ser introducido de forma progresiva añadiendo elementos mediante el botón + o eliminarlos mediante el botón -. También se puede introducir un valor numérico haciendo clic dentro de la caja y escribiendo el valor requerido.



Figura 47 Botones edición número de iconos del menú

Los valores válidos para este campo se encuentran comprendidos entre 1 y 99, ambos inclusive.

3. Título en los iconos

El valor de este parámetro determina si el nuevo menú a crear va a disponer de un campo debajo de cada icono a modo de título.



Figura 48 Opción de título en iconos

4. Disposición de los iconos

Este campo determina la posición de los iconos dentro del nuevo menú, de forma que se pueden establecer en 3 formas diferentes:

- Cuadrícula: permite cualquier valor en forma de matriz. Ejemplos: 2x3, 4x3, 5x5, etc.
- Línea: permite cualquier valor en forma de línea. Ejemplos: 1x3, 1x4 ,1x7, etc.
- Lista: permite cualquier valor en forma de lista. Ejemplos: 3x1, 4x1 ,7x1, etc.

Una interfaz de usuario para configurar la disposición de los iconos. Incluye un menú desplegable etiquetado como 'Tipo de disposición:' con 'Cuadrícula' seleccionada. Debajo hay dos campos de entrada: 'Filas' y 'Columnas'.

Figura 49 Selector disposición iconos del menú

El número de filas y columnas introducidas debe ser suficientes para que los iconos se puedan disponer, pero no de forma que sobren filas o columnas vacías. De no cumplirse alguna de estas condiciones se muestra un mensaje de error indicando si sobran/faltan columnas y filas/iconos.



Figura 50 Mensaje de error del menú

5. Posición de los iconos

Este campo habilita o no la posibilidad de que, **en pantallas posteriores**, se puedan establecer posiciones personalizadas para los iconos del menú. Si se indica que la posición sea automática los iconos se dispondrán de forma equidistante unos de otros.

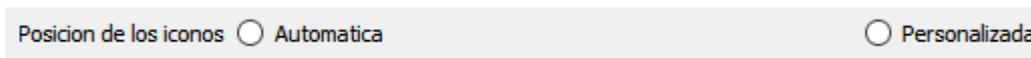
Una interfaz de usuario para seleccionar la posición de los iconos. Incluye el texto 'Posicion de los iconos' seguido de dos botones de radio: 'Automatica' (seleccionado) y 'Personalizada'.

Figura 51 Selector posición de los iconos

Parámetros de cada icono

La siguiente pantalla personaliza los elementos relacionados con los iconos. Para ello, se muestra, por cada icono que se desea crear, una ventana de este tipo. Los parámetros que se pueden personalizar en esta ventana depende de las opciones seleccionadas en la ventana anterior. El único parámetro que obligatoriamente se debe elegir es la imagen del icono. En la parte inferior de la ventana se muestra una barra de progreso para que el usuario pueda determinar en qué punto del proceso se encuentra.

1. Selección de icono



Figura 52 Botón seleccionar icono menú

Para poder seleccionar la imagen del icono, el usuario debe hacer clic en el icono de selección. Una vez lo pulse se abrirá un selector de ficheros en el que el usuario solamente podrá seleccionar imágenes. Una vez seleccione una imagen válida se le mostrará en la ventana la imagen. Si la imagen no es válida o se genera algún problema se le mostrará un mensaje indicándolo.



Figura 53 Mensaje de error imagen del icono

2. Título del icono

En el caso de que en la ventana de [Parámetros Generales](#) se haya seleccionado la opción de “Título de los iconos” como afirmativa, el usuario deberá introducir un título o nombre para el icono. En el caso de que el usuario haya seleccionado la opción como negativa, no se mostrará esta opción.

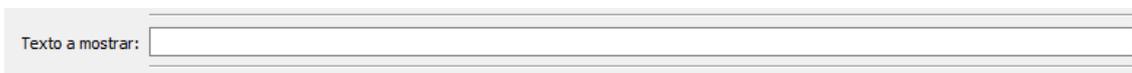


Figura 54 Campo título del icono

3. Posición del icono

En el caso de que en la ventana de [Parámetros Generales](#) se haya seleccionado la opción de “Posición de los iconos” como personalizada, el usuario deberá determinar la posición de cada icono. Esta modificación se realiza en píxeles a partir de la posición inicial, entendiendo posición inicial como la posición que ocupa el icono dentro de la interfaz en la que todos los elementos se encuentran equidistantes entre sí. En el caso de que se haya seleccionado la opción de posición automática no se mostrará esta opción.

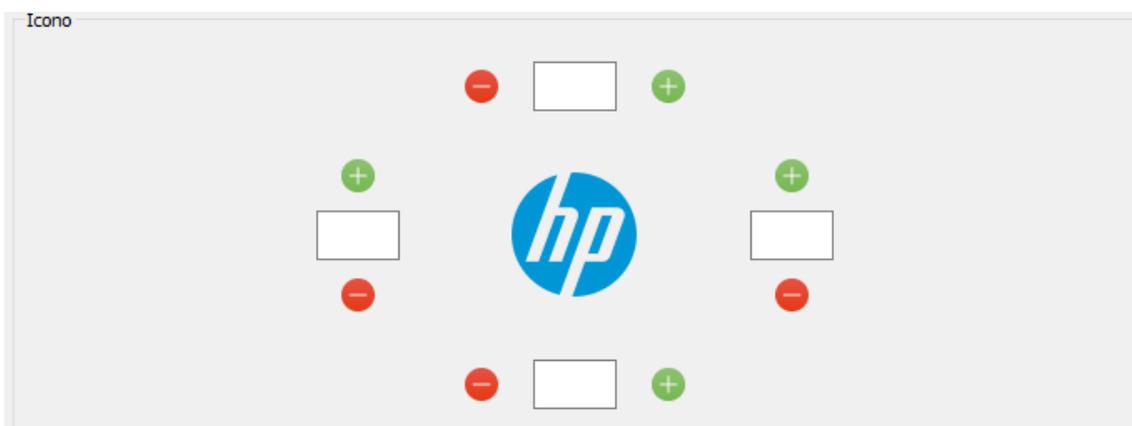


Figura 55 Posición del icono

10.4.2. Información

Esta opción permite al usuario crear interfaces de tipo información. Para hacer uso de esta funcionalidad el usuario debe pulsar el icono de Pantalla de información que se encuentra en la ventana principal de la aplicación.

Cada tarjeta se encuentra formada por un título, unos campos de información, cada uno con su correspondiente valor y, opcionalmente con un icono.

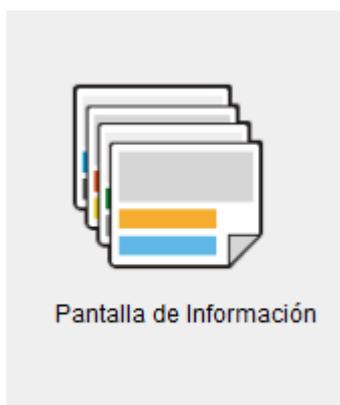


Figura 56 Icono creación interfaz información

Para poder realizar este proceso se deben establecer una serie de parámetros, estos se dividen en dos tipos:

- Parámetros generales de la interfaz: estos parámetros son comunes a la interfaz, como el tamaño, el fondo, el número de iconos del menú, etc.
- Parámetros de cada tarjeta: estos son personalizables para cada tarjeta de información dentro de la interfaz, como el icono, el número de propiedades, el valor de cada propiedad, etc.

Una vez se encuentren rellenos todos los datos de cada ventana el usuario podrá continuar con el siguiente paso en el proceso.

Parámetros generales de la pantalla de información

1. Tamaño y fondo

En este apartado agrupamos los dos primeros parámetros dado que ambos afectan a la totalidad de la interfaz, no a elementos dentro de ella.



Figura 57 Propiedades generales interfaz información

Las dimensiones de la nueva interfaz vienen definidas por su alto y su ancho. Ambos valores se introducen en píxeles.

Para elegir el fondo de la interfaz disponemos de la opción de elegir una imagen o un color. En la primera opción una vez hagamos clic en el icono se nos abre un dialogo para poder elegir el fichero a utilizar entre todos los que se encuentren en el sistema. Si se hace clic en la opción de color se nos abre un dialogo en el que se nos muestra una paleta en la que podremos seleccionar el color deseado.



Figura 58 Botones selección de fondo interfaz información

Una vez completemos una de las dos opciones y si todo ha ido bien, se nos mostrará un mensaje indicando que el fondo se ha establecido correctamente. En el caso de que se encuentre algún error, se mostrará un mensaje al usuario indicándole el problema.

Para modificar el fondo establecido únicamente es necesario hacer clic en una de las dos opciones y seleccionar el nuevo fondo.

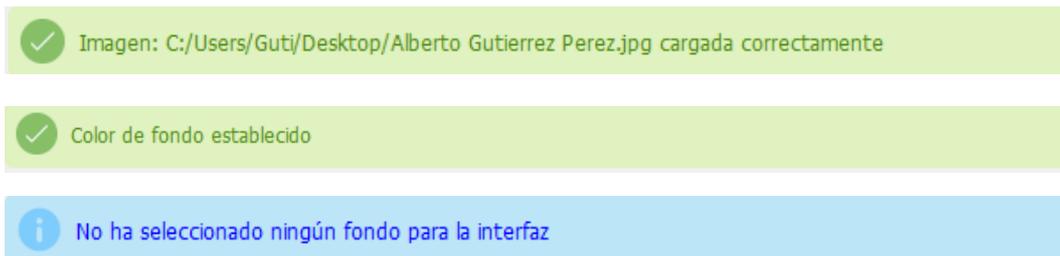


Figura 59 Mensajes información interfaz información

2. Número de elementos

Este parámetro establece el número de tarjetas con las que va a contar la interfaz.



Figura 60 Edición número de elementos interfaz información

El parámetro puede ser introducido de forma progresiva añadiendo elementos mediante el botón + o eliminarlos mediante el botón -. También se puede introducir un valor numérico haciendo clic dentro de la caja y escribiendo el valor requerido.



Figura 61 Botones edición número de elementos información

Los valores válidos para este campo se encuentran comprendidos entre 1 y 99, ambos inclusive.

3. Tipo de disposición

Este campo determina la posición de los iconos dentro de la nueva interfaz de información, de forma que se pueden establecer en 3 formas diferentes:

- Cuadrícula: permite cualquier valor en forma de matriz. Ejemplos: 2x3, 4x3, 5x5, etc.
- Línea: permite cualquier valor en forma de línea. Ejemplos: 1x3, 1x4 ,1x7, etc.
- Lista: permite cualquier valor en forma de lista. Ejemplos: 3x1, 4x1 ,7x1, etc.

Figura 62 Selector disposición tarjetas

El número de filas y columnas introducidas debe ser suficientes para que los iconos se puedan disponer, pero no de forma que sobren filas o columnas vacías. De no cumplirse alguna de estas condiciones se muestra un mensaje de error indicando si sobran/faltan columnas y filas/iconos.

 Para el número de iconos actual sobran filas/columnas

Figura 63 Mensaje de error información

4. Iconos

Este campo habilita o no la posibilidad de que, **en pantallas posteriores**, se pueda establecer un icono para cada tarjeta dentro de la interfaz de información.

Figura 64 Selector iconos información

5. Disposición dentro de la tarjeta

En este campo el usuario deberá seleccionar la disposición de los campos de información dentro de cada tarjeta. Al tratarse de una disposición los valores posibles son los mismos que para la disposición general de la interfaz:

- Matriz
- Lista
- Línea

Figura 65 Selector disposición propiedades tarjeta

Parámetros de cada tarjeta

La siguiente pantalla personaliza los elementos relacionados con cada una de las tarjetas de la interfaz. Para ello, se muestra, por cada tarjeta que se desea crear, una ventana de este tipo. Los parámetros que se pueden personalizar en esta ventana depende de las opciones seleccionadas en la ventana anterior.

1. Título de la tarjeta

En este campo el usuario establece el valor del título de la tarjeta, el cual va separado del resto de la información de la tarjeta de forma que se pueda identificar fácilmente.

Un formulario de entrada de texto con el label "Titulo Tarjeta" a la izquierda y un campo de entrada rectangular a la derecha.

Figura 66 Título tarjeta

2. Icono

Establece el icono dentro de la interfaz. Para ello el usuario debe pulsar el icono de selección. Una vez lo haya pulsado el sistema le mostrará un selector de ficheros donde el usuario debe seleccionar la imagen que desea utilizar como icono.



Figura 67. Botón seleccionar imagen

En el caso de que la imagen sea válida se muestra un mensaje indicando al usuario que la imagen se ha establecido correctamente.



Figura 68 Icono añadido información

Por el contrario, si la imagen no es válida se muestra un mensaje de error indicando que debe seleccionar una imagen válida para poder seguir con el proceso.



Figura 69 Mensaje de error tarjeta

3. Propiedades y su valor

Una vez el usuario ha establecido la disposición de los elementos dentro de la tarjeta, la aplicación le permite establecer las propiedades y sus valores. Para ello la aplicación dispone una pequeña herramienta en la que el usuario puede ver, crear, editar y eliminar propiedades.

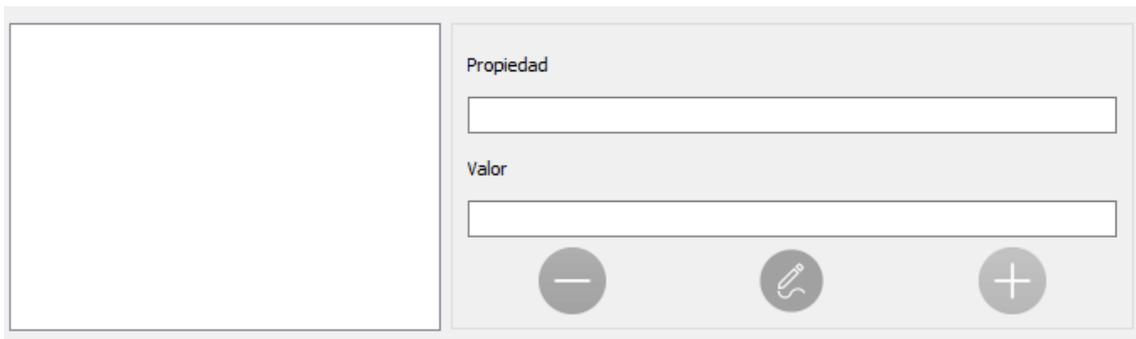


Figura 70 Propiedades tarjeta

Esta herramienta dispone de dos partes, la parte de visualización en la que se muestran las propiedades que el usuario ya ha creado, y la parte de creación en la que el usuario.

Una vez el usuario ha introducido el nombre de una propiedad y su valor, la aplicación le permite crearla pulsando en el botón de creación.



Figura 71
Añadir propiedad
tarjeta

En ese momento, la propiedad se guarda y se añade a la lista de propiedades para que el usuario pueda observar que propiedades ha añadido. Si se pulsa en una de las propiedades añadidas la aplicación permite al usuario modificar su valor y actualizarlo pulsando el botón de edición.



Figura 72 Editar propiedad tarjeta

En caso de que el usuario desee eliminar una propiedad ya añadida debe seleccionarla y pulsar el botón de eliminar.



*Figura 73
Eliminar propiedad
tarjeta*

10.4.3. Tinta

Esta opción permite al usuario crear interfaces de tipo nivel de tinta. Para hacer uso de esta funcionalidad el usuario debe pulsar el icono de tintas que se encuentra en la ventana principal de la aplicación.

Para poder realizar este proceso se deben establecer una serie de parámetros, entre los que se encuentran:

- Tamaño de la interfaz, número de niveles, color de los niveles, disposición, información a mostrar sobre la tinta, etc.

Una vez se encuentren rellenos todos los datos de cada ventana el usuario podrá continuar con el siguiente paso en el proceso.



*Figura 74 Icono creación
interfaz tinta*

1. Tamaño

Las dimensiones de la nueva interfaz vienen definidas por su alto y su ancho. Ambos valores se introducen en pixeles.

Tamaño: Alto	<input type="text" value="0"/>	Ancho	<input type="text" value="0"/>
--------------	--------------------------------	-------	--------------------------------

Figura 75 Tamaño interfaz tinta

2. Número de elementos

Este parámetro establece el número de niveles de tinta con los que va a contar la interfaz.

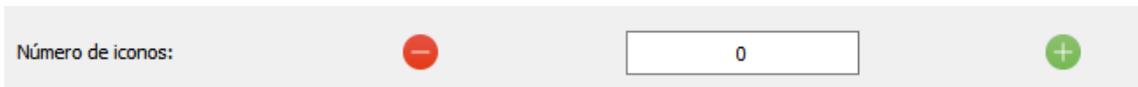


Figura 76 Edición número de elementos interfaz tinta

El parámetro puede ser introducido de forma progresiva añadiendo elementos mediante el botón o eliminarlos mediante el botón. También se puede introducir un valor numérico haciendo clic dentro de la caja y escribiendo el valor requerido. Los valores válidos para este campo se encuentran comprendidos entre 1 y 99, ambos inclusive.

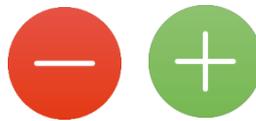


Figura 77 Botones edición número de elementos interfaz tinta

3. Tipo de disposición

Este campo determina la posición de los niveles dentro de la nueva interfaz, de forma que se pueden establecer en 3 formas diferentes:

- Cuadrícula: permite cualquier valor en forma de matriz. Ejemplos: 2x3, 4x3, 5x5, etc.
- Línea: permite cualquier valor en forma de línea. Ejemplos: 1x3, 1x4 ,1x7, etc.
- Lista: permite cualquier valor en forma de lista. Ejemplos: 3x1, 4x1 ,7x1, etc.



Figura 78 Selector disposición elementos tinta

El número de filas y columnas introducidas debe ser suficientes para que los elementos se puedan disponer, pero no de forma que sobren filas o columnas vacías. De no cumplirse alguna de estas condiciones se muestra un mensaje de error indicando si sobran/faltan columnas y filas/iconos.



Figura 79 Mensaje de error interfaz tinta

4. Selección de colores

En este apartado el usuario selecciona los diferentes colores para cada nivel de tinta. Para ello, dispone de un botón por cada nivel, el cual, tras pulsarlo, le mostrará una paleta de colores para que seleccione el que desea. Si el color es válido el botón correspondiente a ese nivel de tinta se vuelve del color seleccionado.

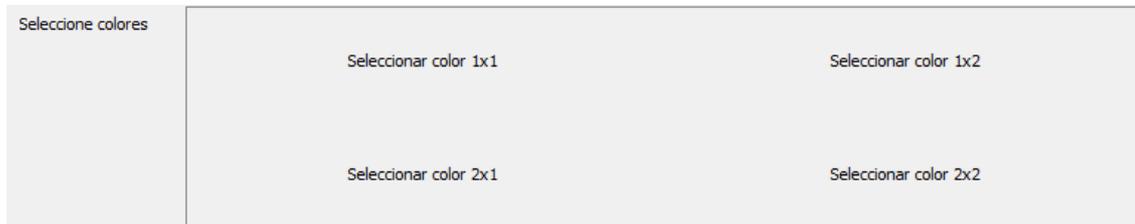


Figura 80 Selector colores niveles de tinta

5. Información a mostrar

Al tratarse de una interfaz de tipo gráfica, el usuario puede determinar la forma de disponer la información sobre los niveles de tinta. Entre las opciones a elegir se encuentran:

- Porcentaje: se muestra el nivel de tinta de la forma: 40%
- Número: se muestra el nivel de tinta de la forma: 40.
- Nada: no se muestra información adicional sobre el nivel de la tinta.

10.5. Edición

Gracias a esta funcionalidad el usuario puede editar una interfaz ya creada con anterioridad por la aplicación. Para editar una interfaz el usuario debe pulsar sobre el icono de edición que aparece en la ventana principal de la aplicación.

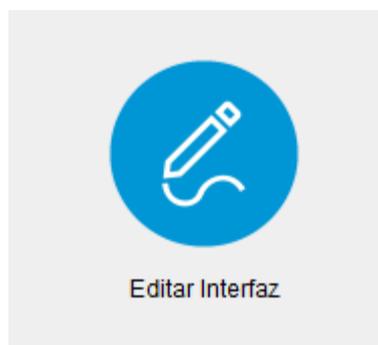


Figura 81 Icono editar interfaz

Una vez que el usuario ha pulsado sobre el botón, la aplicación le muestra una ventana donde se le indica que puede arrastrar y soltar un fichero para editar o introducir la ruta del fichero mediante un selector de ficheros. Ambos métodos únicamente aceptan ficheros QML.

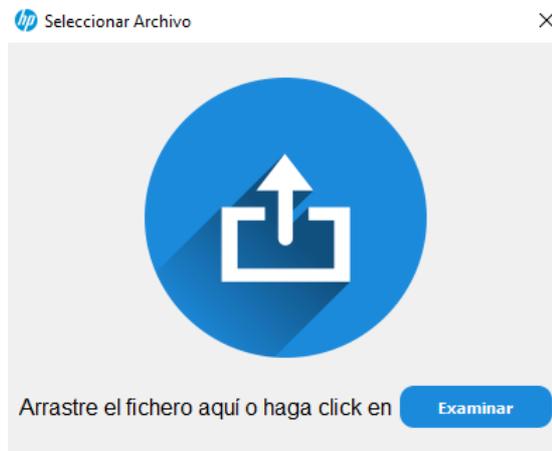


Figura 82 Ventana seleccionar fichero a editar

En el caso de que el usuario arrastre un fichero sobre la ventana y este no se trate de un fichero con formato QML se le mostrará un error, en el caso del selector únicamente son seleccionables ficheros con la extensión correcta.



Figura 83 Error de formato en el archivo a editar

Tanto si el usuario arrastra un fichero con extensión QML como si examina la ruta del archivo a editar, la aplicación, en función del tipo de interfaz, abre el apartado de creación con la información de la interfaz a editar.

En el caso de que exista algún problema con la interfaz a editar, se muestra un mensaje de error indicando el elemento de la interfaz y el motivo del mismo.

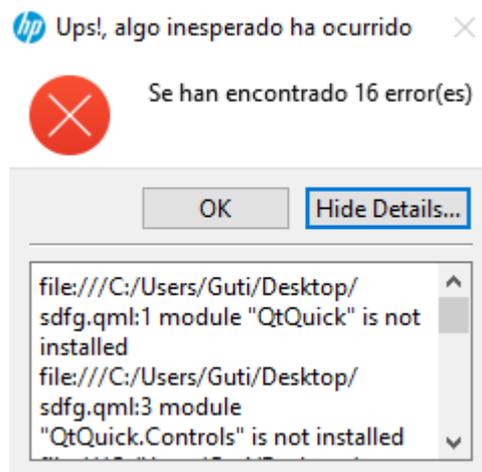


Figura 84 Error en el fichero QML a editar

10.6. Barra de “Menú”

10.6.1. Home

Esta opción otorga la posibilidad de volver a la ventana principal de la aplicación desde cualquier punto de la misma. En el caso de que el usuario haga clic en este icono durante la creación o edición de una interfaz, se le mostrará un mensaje de confirmación dado que una vez que se encuentre en la ventana principal no se podrá recuperar el trabajo realizado hasta el momento previo al hacer clic. De esta forma se impide que se pierda trabajo de forma involuntaria.

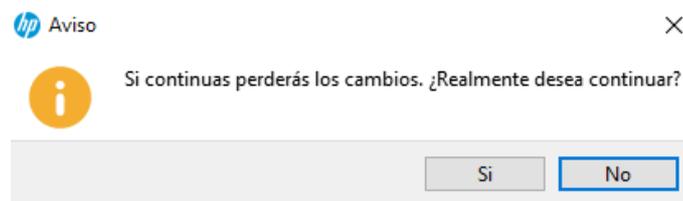


Figura 85 Confirmación volver a inicio

10.6.2. Ajustes

La ventana de ajustes permite al usuario realizar diferentes personalizaciones sobre la propia aplicación. Entre ellas se encuentran:

1. La posibilidad de establecer una ruta donde guardar los ficheros por defecto en vez de preguntar por una ubicación cada vez que el usuario finalice la creación de una interfaz.

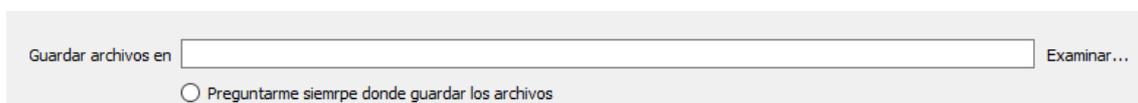


Figura 86 Directorio por defecto

2. Posibilidad de cambiar el idioma de la aplicación: la aplicación dispone de dos idiomas, inglés y español. Dependiendo de que opción seleccione el usuario se

modificará el idioma de toda la aplicación. Por defecto la aplicación se encuentra en español.



Figura 87 Idioma de la aplicación

3. Restablecer los valores por defecto de la aplicación: Para ello el usuario debe hacer clic en el icono de reestablecer, y una vez confirme la acción se mostrará un mensaje de que se han reestablecido los valores predeterminados.



Figura 88 Restaurar valores

4. Ver la información asociada a la aplicación: en este apartado el usuario puede obtener información sobre la aplicación como su versión, su desarrollador, etc.

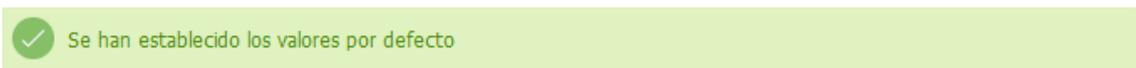


Figura 89 Mensaje de éxito ajustes

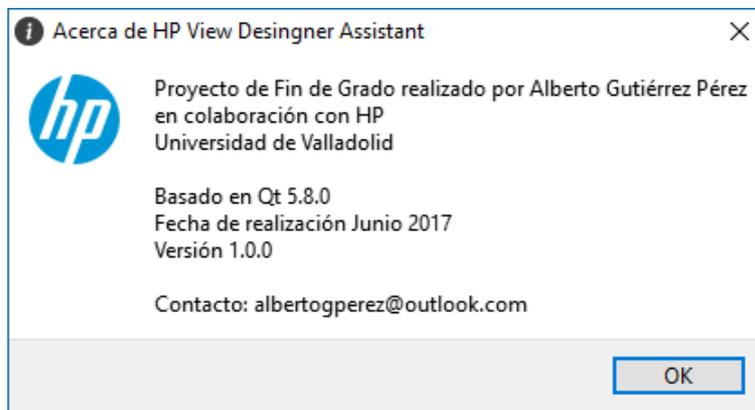


Figura 90 Ventana Acerca de

10.6.3. Ayuda

Esta opción permite al usuario obtener ayuda sobre cualquier punto de la aplicación. Para ello solamente debe hacer clic sobre el siguiente icono:



Figura 91
Icono de ayuda

En ese momento la aplicación mostrará en el navegador por defecto del sistema la ayuda del punto en el que se encuentre actualmente el usuario. Al tratarse de una ayuda en formato web, el usuario puede navegar entre las diferentes páginas de ayuda.

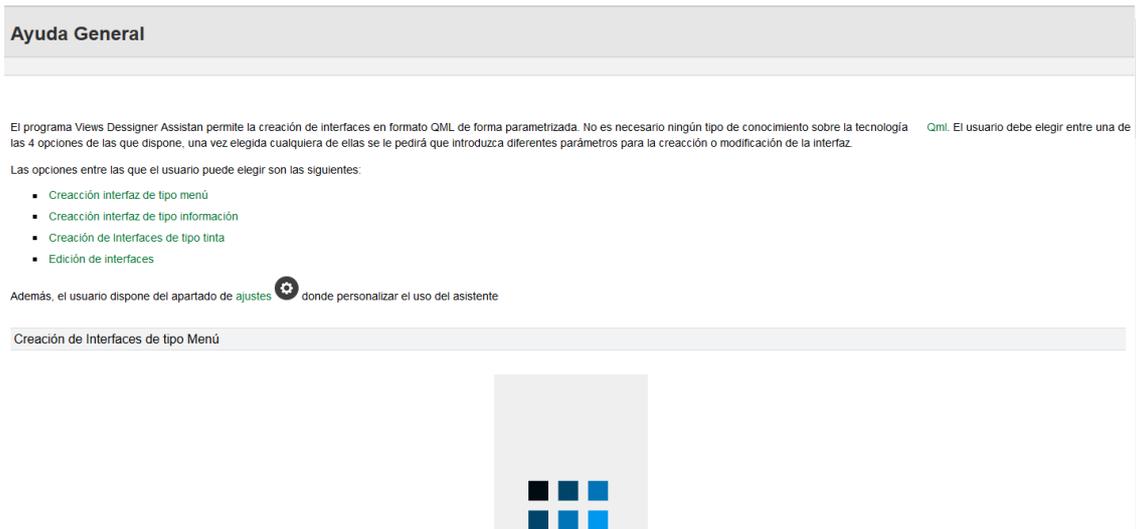


Figura 92 Ventana principal de ayuda

Capítulo 11. Conclusiones

Al finalizar el desarrollo del proyecto se puede afirmar que se han completado todos los objetivos propuestos. Igualmente, se han cumplido los requisitos establecidos al comienzo del proyecto.

La implementación de un asistente al estilo "Wizard" que permita al usuario-programador parametrizar una interfaz, la cual debe generarse en formato QML. Para cualquiera de los tres tipos de interfaz que se generan, se permite personalizar la totalidad de la interfaz. Adicionalmente, se han introducido diferentes funcionalidades para mejorar la interacción con el usuario como son; la posibilidad de, a la hora de editar una interfaz, "arrastrar y soltar" ficheros o la oportunidad de poder establecer un directorio de trabajo predeterminado.

Gracias a la realización de este proyecto se han aprendido conocimientos sobre QML y Qt su potencial y su capacidad de adaptación. Asimismo, se ha ampliado de forma considerable el conocimiento sobre la programación orientada a objetos, en concreto sobre la programación en C++.

A nivel organizativo, se ha observado la complejidad de monitorizar y gestionar un proyecto de dimensiones considerables. La administración de los recursos y la delimitación del alcance han sido cruciales para poder conseguir finalizar el proyecto.

11.1. Líneas futuras

Tras finalizar por completo el desarrollo del proyecto se establecen posibles mejoras con el fin de perfeccionar y aumentar el alcance de la aplicación. Al tratarse de un generador de interfaces, la mejora la interacción con el usuario y la personalización, son los principales objetivos. La base que permite la mejora de estas características es la retroalimentación por parte del usuario final, lo cual en muchas ocasiones no se puede obtener de forma continuada.

Por otro lado, las continuas mejoras en las versiones de Qt y QML hace que la tarea de mantenimiento y actualización de la aplicación se conviertan en otras posibles mejoras continuas de la aplicación.

A continuación, vamos a enumerar una serie de posibles mejoras que se podrían implementar en futuros desarrollos:

- Interfaces QML con scroll: de esta forma, tras fijar el tamaño de la interfaz se pueden introducir cuantos elementos requiera el usuario sin que afecte a su tamaño visual dentro de la interfaz
- Interfaces QML con cabecera o barra de navegación: aceptar que el usuario pueda elegir entre una interfaz con barra de navegación o no. De esta forma se facilita el uso de las interfaces en sistemas complejos.
- Posición relativa de los elementos dentro de las interfaces QML: permitir al usuario posicionar los elementos en función del resto de componentes de la interfaz.
- Detección automática del tamaño de la pantalla: permitir que la interfaz se ajuste automáticamente a la pantalla en la que es encuentre. De esta forma, no es necesario que el usuario conozca las dimensiones del sistema donde se va a utilizar la interfaz.

- Plantillas favoritas: permitir al usuario establecer unos valores predeterminados para los tres tipos de interfaz de forma que se agilice la creación masiva de interfaces similares.
- Parametrización personalizada: permitir al usuario personalizar más parámetros de la interfaz de forma dinámica. De forma que el nivel de personalización de una interfaz la decida el propio usuario.
- Añadir otros tipos de interfaz: permitir al usuario añadir otros tipos de interfaz a generar.

Bibliografía

-
- [1.] Qt Biblioteca [Online] Enlace: [https://es.wikipedia.org/wiki/Qt_\(biblioteca\)](https://es.wikipedia.org/wiki/Qt_(biblioteca)) [Accedido: 22-abril-2017]
- [2.] Qt Software [Online] Enlace: [https://en.wikipedia.org/wiki/Qt_\(software\)](https://en.wikipedia.org/wiki/Qt_(software)) [Accedido: 21-enero-2017]
- [3.] What is PyQt [Online] Disponible en: <https://riverbankcomputing.com/software/pyqt/intro> [Accedido: 21-enero-2017]
- [4.] ¿Qué es GNU? [Online] Enlace: <https://www.gnu.org/> [Accedido: 6-febrero-2017]
- [5.] GNU Lesser General Public License [Online] Enlace: https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License [Accedido: 6-febrero-2017]
- [6.] Qt Documentation [Online] Enlace: <http://doc.qt.io/qt-5/qmlapplications.html> [Accedido: 10-febrero-2017]
- [7.] Autor: Lee Zhi Eng Título: Qt5 C++ GUI Programming Cookbook Editorial: Pack Publishing, 2016.
- [8.] Qt Tools and Versions [Online] Enlace: https://wiki.qt.io/Qt_5.3_Tools_and_Versions [Accedido: 19-mayo-2017]
- [9.] QML Wikipedia [Online] Enlace: <https://es.wikipedia.org/wiki/QML> [Accedido: 3-marzo-2017]
- [10.] QML Applications [Online] Enlace: <http://doc.qt.io/qt-5/qmlapplications.html> [Accedido: 12-febrero-2017]
- [11.] Models and Views in Qt Quick [Online] Enlace: <http://doc.qt.io/qt-5/qtquick-modelviewsdata-modelview.html> [Accedido: 20-febrero-2017]
- [12.] Using Qt Quick Designer [Online] Enlace: <http://doc.qt.io/qtcreator/creator-using-qt-quick-designer.html> [Accedido: 19-mayo-2017]
- [13.] Creating a Qt Quick Application [Online] Enlace: <http://doc.qt.io/qtcreator/qtcreator-transitions-example.html> [Accedido: 19-mayo-2017]
- [14.] Dynamic QML Object Creation from JavaScript [Online] Enlace: <http://doc.qt.io/qt-5/qtqml-javascript-dynamicobjectcreation.html> [Accedido: 25-mayo-2017]
- [15.] Apuntes de PGP. Tema 3 Proceso unificado. Curso 16/17 [En línea] Enlace: https://aulas.inf.uva.es/pluginfile.php/26288/mod_resource/content/6/PGP_plan_UP_1617.pdf [Ultima conexión: 11 Diciembre 2016]
- [16.] Apuntes de PGP. Gestión de riesgos. Pablo de la Fuente Redondo. Curso 16/17 [En línea] Enlace: https://aulas.inf.uva.es/pluginfile.php/26242/mod_resource/content/4/PGP_riesgos_1617.pdf [Ultima conexión: 11 Diciembre 2016]

- [17.] Gestión de riesgos INCIBE[Online] Enlace: https://www.incibe.es/extfrontinteco/img/File/empresas/guias/Guia_gestion_riesgos/guiagestionriesgos.pdf [Accedido: 6-mayo-2017]
- [18.] OMG Unified Modeling Language [Online] Enlace: <http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF> [Accedido: 10-mayo-2017]
- [19.] QMainWindow Class [Online] Enlace: <http://doc.qt.io/qt-5/qmainwindow.html> [Accedido: 2-abril-2017]
- [20.] QPushButton [Online] Enlace: <http://doc.qt.io/qt-5/qpushbutton.html> [Accedido: 21-marzo-2017]
- [21.] QLineEdit Class [Online] Enlace: <http://doc.qt.io/qt-5/qlineedit.html> [Accedido: 21-marzo-2017]
- [22.] QLayout Class [Online] Enlace: <http://doc.qt.io/qt-5/qlayout.html> [Accedido: 21- marzo -2017]
- [23.] QHBoxLayout [Online] Enlace: <http://doc.qt.io/qt-5/qhboxlayout.html> [Accedido: 21- marzo -2017]
- [24.] QVBoxLayout [Online] Enlace: <http://doc.qt.io/qt-5/qvboxlayout.html> [Accedido: 21- marzo -2017]
- [25.] QComboBox Class [Online] Enlace: <http://doc.qt.io/qt-5/qcombobox.html> [Accedido: 2- abril -2017]
- [26.] QRadioButton Class [Online] Enlace: <http://doc.qt.io/qt-5/qradiobutton.html> [Accedido: 2- abril -2017]
- [27.] QLabel Class[Online] Enlace: <http://doc.qt.io/qt-5/qlabel.html> [Accedido: 27- marzo -2017]
- [28.] QMessageBox Class[Online] Enlace: <http://doc.qt.io/qt-5/qmessagebox.html> [Accedido: 22-marzo -2017]
- [29.] QFileDialog Class [Online] Enlace: <http://doc.qt.io/qt-5/qfiledialog.html> [Accedido: 20- marzo -2017]
- [30.] QColorDialog Class [Online] Enlace: <http://doc.qt.io/qt-5/qcolordialog.html> [Accedido: 25- marzo -2017]
- [31.] QML Documents [Online] Enlace: <http://doc.qt.io/qt-5/qtqml-documents-topic.html> [Accedido: 13-mayo -2017]
- [32.] Interacting with QML Objects from C++ [Online] Enlace: <http://doc.qt.io/qt-5/qtqml-cppintegration-interactqmlfromcpp.html> [Accedido: 28-mayo-2017]
- [33.] The Qt Resource System [Online] Enlace: <http://doc.qt.io/qt-5/resources.html> [Accedido: 17-mayo-2017]
- [34.] QToolTip [Online] Enlace: <http://doc.qt.io/qt-5/qtooltip.html> [Accedido: 19-marzo-2017]
- [35.] QFontMetrics [Online] Enlace: <http://doc.qt.io/qt-5/qfontmetrics.html> [Accedido: 5-marzo-2017]
- [36.] QSettings [Online] Enlace: <http://doc.qt.io/qt-5/qsettings.html> [Accedido: 29-abril-2017]
- [37.] QVariant [Online] Enlace: <http://doc.qt.io/qt-5/qvariant.html> [Accedido: 3-mayo-2017]

-
- [38.] Image Provider QML [Online] Enlace: <http://www.lothlorien.com/kf6gpe/?p=234> [Accedido: 28-marzo-2017]
- [39.] QTextEdit [Online] Enlace: <http://doc.qt.io/qt-5/qtextedit.html> [Accedido: 3-abril-2017]
- [40.] QTextBrowser [Online] Enlace: <http://doc.qt.io/qt-5/qtextbrowser.html#details> [Accedido: 15-abril-2017]
- [41.] QDesktop Services [Online] Enlace: <http://doc.qt.io/qt-5/qdesktopservices.html#details> [Accedido: 15-abril-2017]
- [42.] QAbstract Item Model Class [Online] Enlace: <http://doc.qt.io/qt-5/qabstractitemmodel.html#details> [Accedido: 21-abril-2017]
- [43.] The Property System [Online] Enlace: <http://doc.qt.io/qt-5/properties.html> [Accedido: 2-mayo-2017]
- [44.] QMetaObject Class [Online] Enlace: <http://doc.qt.io/qt-5/qmetaobject.html> [Accedido: 21-abril-2017]
- [45.] Bug Reports 52618 [Online] Enlace: <https://bugreports.qt.io/browse/QTBUG-52618> [Accedido: 1-junio-2017]
- [46.] ISO C++ [Online] Enlace: <https://isocpp.org/faq> [Accedido: 23-mayo-2017]
- [47.] C++ Wikipedia [Online] Enlace: <https://en.wikipedia.org/wiki/C%2B%2B> [Accedido: 23-mayo-2017]
- [48.] Qt Creator [Online] Enlace: https://wiki.qt.io/Qt_Creator/es [Accedido: 26-abril-2017]
- [49.] Qt [Online] Enlace: <https://www.qt.io/> [Accedido: 9-junio-2017]

Anexos

Anexo I – Informes de progreso

Hito 1. 10/02/2017 – 24/02/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 1. Además, se plantean diferentes dudas sobre cómo se espera que funcione la aplicación. Hay que destacar que ya se han pensado diferentes mejoras y añadidos para mejorar la funcionalidad y manejo de la aplicación que se añadirán a la misma siempre y cuando el factor temporal lo permita.

a. Detalles generales

En este apartado se van a aclarar y explicar diferentes detalles del diseño de la aplicación que se verán modificados una vez se comience con el desarrollo de la lógica de la misma.

- La aplicación dispone de tres botones en la barra inferior en los cuales se muestra el acceso al menú principal, a los ajustes y a la ayuda. En este momento si el usuario hace clic la aplicación le redirige a dicho sitio, posteriormente se añadirá un pop-up de aviso al hacer clic en los botones del menú de inicio y de ajustes avisando de que va a ir a una ventana diferente. Dicho mensaje solamente se mostrará si el usuario se encuentra realizando otra acción en la aplicación, como por ejemplo a mitad de proceso en la creación de una nueva interfaz.

-

b. Pantalla Principal

En la pantalla principal se ha utilizado el diseño y la iconografía de alguna de las imágenes prototipo que se proporcionaron como documentación. Se ha realizado de dicha forma dado que el desarrollo del producto se centra en funcionalidad y no tanto en diseños gráficos.

c. Pantalla de ajustes

La pantalla cuenta únicamente con dos tipos de ajustes:

- Una dirección por defecto para guardar los documentos. (o la opción de que se pregunte siempre)
- Una opción de cambio de idioma. En este apartado hay que informar de que se ha buscado la opción de traducir los textos y parece que puede ser implementada tras realizar una investigación de su funcionamiento. No parece un requisito funcional de importancia por lo que se realizará siempre que el resto de la funcionalidad se encuentre completada dentro de los plazos previstos.

Dispone también de un botón “Acerca de” para que se pueda conocer el ámbito y la autoría del proyecto. Ahora mismo dispone de un mensaje descriptivo que hay que extender y modificar.

En esta pantalla se puede probar la funcionalidad básica en todos los elementos menos en el apartado de cambio de idioma, puesto que se desarrollará en función del tiempo disponible.

d. Pantalla/botón Ayuda

Este botón muestra diferente información en función de donde nos encontremos. En la situación actual del proyecto solamente muestra un mensaje de información genérico dado que las pantallas y

acciones sobre las mismas aún no se encuentran cerradas. Una vez se concrete más y se establezca el diseño se procederá a redactar los diferentes textos informativos.

e. Creación Menús

Esta acción dispone de dos ventanas:

- La primera en la que se muestran las opciones generales del menú
- Una segunda en la que se establecen los parámetros de cada elemento del menú por lo que esta ventana aparecerá tantas veces como elementos deseemos crear dentro del propio menú

i. Pantalla general

En esta pantalla se establecen los siguientes parámetros:

- Número de elementos: número de iconos que tendrá el menú. Se realizará un combo box personalizado dado que el incremento que proporciona la plataforma no es amigable para el usuario.
- Opción de texto debajo de cada icono.
- Tipo de disposición: despegable que muestra las opciones en función de la elección del usuario.
- Posición de los iconos: esta opción establece si la posición de los iconos la realiza de forma automática la aplicación o el usuario desea personalizar la posición de cada uno. En el caso de que seleccione automática también se le pregunta si desea que los iconos se encuentren con margen entre ellos o no. La opción personalizada deberá personalizar cada elemento por separado en su ventana.

Una vez se encuentren rellenos todos ellos se da la posibilidad de avanzar al usuario. Existe una modificación pendiente de la interfaz, de forma que los tres iconos de la barra inferior se conviertan en un único botón de “atrás” una vez se muestre el botón de continuar.

ii. Pantalla de cada elemento

Esta ventana dispone de las siguientes opciones:

- Opción de seleccionar un icono, el cual abre un menú browser típico en búsqueda de fichero de tipo imagen.
- Texto a mostrar: dialogo que se elimina si el usuario ha seleccionado que no quiere texto debajo de cada icono.
- Posición del icono: esta opción solamente se encontraría disponible si el usuario ha seleccionado que la posición de los iconos es personalizada. No se encuentra implementado puesto que aún no se ha decidido que parámetros se van a solicitar para posicionar los elementos.

Como se puede observar posee una barra de progreso en la parte inferior de la pantalla, dicha barra se ha establecido para indicar al usuario el progreso de creación que lleva dentro de todos los iconos, de forma que sepa cuanto le falta para su finalización. No se trata de un diseño final, pero se ha concretado que se debe mostrar algún tipo de progreso dado que todas las ventanas específicas de elementos del menú son idénticas y el usuario puede pensar que la aplicación no avanza.

Tal y como se encuentra la aplicación en este momento esta ventana no permite avanzar más puesto que el siguiente paso sería guardar el fichero.

f. Creación Pantalla Información

i. Pantalla general

En el caso de creación de pantallas de tipo información los parámetros solicitados son los siguientes:

- Número de elementos: número de tarjetas de información
- Tipo de disposición: establece como se van a distribuir las tarjetas de información dentro de la vista general.
- Si los elementos van a disponer de iconos o no.

Una vez se rellenen todos los campos se permite continuar.

ii. Pantalla de cada elemento

Igual que para los menús se muestra una pantalla para cada tarjeta. En este caso se solicita el título de la tarjeta y la disposición de los elementos dentro de la tarjeta. Una vez se introduzcan esos datos se mostrarán el resto de campos para rellenar los elementos dentro de la tarjeta, así como la opción de búsqueda de iconos si el usuario ha marcado la opción en la ventana anterior.

En esta pantalla se establecerá también una barra de progreso o alguna forma de indicar la sensación de avance al usuario.

Esta disposición de elementos dentro de una tarjeta se ha realizado pensando en la imagen de pantallas de información proporcionadas en el prototipo:

Printer Information

PRODUCT NAME	HP PageWide XL 8000		
SERIAL NUMBER	CN111111111111	PRODUCT NUMBER	CZ309A
SERVICE ID	0	DATE	11/14/16

g. Creación Pantalla de tintas

Este apartado únicamente dispone de una ventana en la cual se dispone de las siguientes opciones:

- Número de elementos (como ya se ha comentado el diseño deberá cambiarse dado que el asignado por defecto no es usable).
- Selección de colores: en el prototipo consta de 4 botones en los cuales al hacer clic se muestra una paleta de colores que establece el color al botón para que el usuario pueda ver una vista previa. En número de botones se modificará en función del primer parámetro.
- Tipo de disposición de los elementos.
- Información a mostrar con la tinta: si se desea un porcentaje, un número, texto o nada.

h. Edición Interfaces

Para la edición de interfaces se le indica al usuario que seleccione un fichero en formato QML para editar. Una vez seleccione un fichero el programa deberá determinar qué tipo de interfaz es y mostrará al usuario las ventanas explicadas anteriormente con los datos para modificar.

i. Consideraciones

Se ha pensado en más opciones, pero de momento, no se ha concretado ninguna que destaque por falta de funcionalidad en la aplicación. En el caso de que se detecte alguna que considere que se deba incorporar, hágalo saber.

Hito 2. 25/02/2017 – 17/03/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 2. Además, se plantean diferentes dudas sobre cómo se espera que funcione la aplicación.

a. Problemas encontrados

La mayoría de los problemas encontrados han sido debido a que la documentación encontrada se basaba en la edición dinámica de ficheros QML. Tras conseguir la edición mediante esta técnica se observa que estas modificaciones no se pueden hacer permanentes, es decir, no hay forma de obtener el QML generado una vez se modifica.

b. Preguntas

Tanto las imágenes de los iconos como la imagen de fondo son imágenes predeterminadas cargadas dentro del propio programa. Esto es debido a que, para mostrar la vista previa, es necesario que las fotos se encuentren dentro de los recursos del programa. La pregunta se centra en si se mantienen los inputs para que el usuario seleccione el fichero y no se muestre la vista previa o se eliminan los inputs y se muestran las imágenes predeterminadas.

c. Mejoras y posibles evolutivos

Se ha observado en la documentación de Qt que existe la posibilidad de añadir un header o footer de forma sencilla. Esto sería una barra general dentro de los menús en la parte superior o inferior de los mismos. El único impedimento que se ha encontrado hasta ahora es el de generar el QML necesario para ello. Espero poder disponer del tiempo necesario para incluir esta funcionalidad.

Además, se ha pensado en añadir la opción de elegir directamente la posición de un elemento respecto de otro. Ahora mismo está una opción de editar la posición de un elemento, pero de forma general, diciendo la separación entre el propio elemento y su parte superior, inferior, izquierda y derecha

Realizando pruebas se ha encontrado algunos errores menores, esta circunstancia no me preocupa demasiado dado que dentro de la planificación se encuentra un apartado de pruebas.

Hito 3. 18/03/2017 – 29/03/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 3. Además, se enuncian las partes que se van a ver modificadas.

a. Tamaño elementos de las interfaces

A la hora de realizar la creación de una de las interfaces, ya sea menú o de tipo información, el tamaño del icono está prefijado de forma que no se salga de su contenedor. Esta comprobación básica hace que en algunas ocasiones el tamaño del icono sea desproporcionado con respecto al texto que le acompaña. Se ha comprobado en la documentación de Qt que existe la forma de medir el tamaño del texto de tal manera que se pueden realizar ajustes para solucionar esta situación. Esta modificación se realizará previsiblemente para la primera entrega del apartado de pruebas: hito 5 (26/04/2017).

b. Mejoras y posibles evolutivos

Tal y como se dijo en la entrega del hito anterior se ha probado la inserción de footer, es decir, una barra inferior en la interfaz. Se ha probado una versión básica que se puede ver en las capturas adjuntas. Además, se ha cambiado la forma en la que el usuario elige entre los dos tipos de fondos de forma que sea más visual.

Con el fin de ayudar al usuario, aun contando con la ayuda que se encuentra en la parte inferior de la página, se van a añadir algunas ayudas contextuales. Estas nuevas ayudas serán puntuales y aparecerán al mantener el cursor sobre un elemento del asistente.

c. Dudas anteriores

Dado que el uso normal de ficheros QML dentro de una aplicación se realiza de forma que obtenga la mayoría de sus recursos de ficheros de recursos cargados previamente, se ha decidido mostrar al usuario la opción de que seleccione entre los iconos ya cargados en el asistente. La desventaja encontrada con esta solución es que a la hora de utilizar este fichero QML en otra aplicación se debe disponer de una imagen en esa misma ruta puesto que de no ser así no se visualizará imagen alguna. Se ha observado que existe la posibilidad de cargar ficheros de recursos (ficheros de recursos de creados con ayuda de Qt) una vez la aplicación se encuentre en uso, por tanto, se podría añadir la opción de añadir ficheros de recursos externos para que el usuario pueda introducir sus "paquetes de iconos".

d. Comentarios

En el caso de la creación de interfaces de tipo información hay que decir que se deben comprobar algunas navegaciones. Una vez se encuentren añadidas, será difícil la realización de errores irrecuperables por parte del usuario.

Hito 4. 30/03/2017 – 15/04/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 4. Además, se comentan desarrollos que se han realizado de forma complementaria, así como la evolución de los problemas encontrados.

a. Detalles generales

Se ha llevado a cabo la implementación de las interfaces de tipo nivel de tinta. Esta funcionalidad únicamente dispone de una pantalla dado que no se configuran tantos parámetros como en ocasiones anteriores. Uno de estos parámetros es el color de la tinta, esta funcionalidad se ha implementado mediante botones que se disponen en la pantalla de la misma forma que se sitúan posteriormente en el fichero QML. De esta forma el usuario conoce de forma intuitiva como se van a disponer. Aunque no se encuentra finalizado, una vez se seleccione cada color se establece de fondo del botón de forma que se ve que color se ha seleccionado previamente.

b. Otras implementaciones

Además de la implementación planificada se han llevado a cabo otras implementaciones adicionales. Las cuales son:

- Selector de iconos: como se dijo en entregas anteriores se ha implementado un selector que permite elegir entre los iconos previamente cargados en la aplicación. Esta implementación es posible que se suprima puesto que se ha conseguido una forma de establecer cualquier imagen de fondo. Esta forma no se ha probado lo necesario por lo que no se puede asegurar nada, pero lo más posible es que sustituya al selector de iconos.
- Ajustes: se han implementado los ajustes de forma que se guarden en un fichero las características que el usuario decida. No está integrado de forma completa con el sistema, pero ya guarda y restablece los parámetros del usuario.

Hito 5. 17/04/2017 – 26/04/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 5. Además, se comentan los desarrollos realizados durante la duración del hito, aunque no se encontrasen en la planificación inicial.

a. Detalles generales

De forma general se han cumplido las estimaciones realizadas para esta etapa. Se han corregido la totalidad de errores encontrados en la creación de menús, así como los errores producidos al generar interfaces de tipo información. En estas últimas hay que comentar que no se ha llevado a cabo la redimensión del tamaño del texto en función de su longitud y del tamaño de la interfaz. Se espera que en siguiente hito se pueda realizar dicha implementación.

b. Otras implementaciones

Como se comentó en entregas anteriores se ha implementado un sistema para que el usuario pueda elegir cualquier imagen del sistema, sin necesidad de que se encuentre dentro de los recursos del asistente.

Se ha comenzado la implementación de la parte de ayuda dentro del asistente. Esta se encuentra en ficheros HTML para poder manejar de forma cómoda su edición. En una primera implementación el fichero se mostraba dentro de la aplicación dado que Qt soporta tanto HTML como hojas de estilo CSS. Esta implementación se descarta dado que la clase propia de Qt que permite visualizar elementos en formato HTML y CSS no soporta todas funcionalidades de las mismas. Existe la posibilidad de añadir módulos propios de Qt que soporten toda la funcionalidad de las tecnologías web, algunos se encuentran en desuso y son demasiado “pesados” como para utilizarlos únicamente para mostrar un fichero.

Teniendo en cuenta todo lo anterior se realiza una solución que despliega la ayuda ayudándose en el explorador web del sistema donde se esté ejecutando el asistente. Esta funcionalidad de encuentra parcialmente completada, es decir, su implementación se ha desarrollado completamente, aunque faltan por añadir las diferentes ayudas para cada una de las partes del asistente. Actualmente se encuentran las ayudas del menú principal, así como las de creación de menús.

Hito 6. 27/04/2017 – 12/05/2017

El siguiente documento presenta explicaciones de la parte del proyecto entregada en el hito 6. Además, se comentan los desarrollos finalizados que se iniciaron en etapas anteriores.

a. Detalles generales

Durante este hito se han llevado a cabo dos importantes implementaciones. La primera, basada en la redimensión del texto de las interfaces de información, viene arrastrada del hito anterior donde se encontraba planificada. Esta implementación ha consumido una gran cantidad de tiempo, haciendo que no haya sido posible recuperar la planificación original del proyecto.

Como parte positiva hay que decir que se ha realizado una implementación que facilita posteriores ediciones de las interfaces además de agilizar su creación. Esta ventaja se ha obtenido modificando la forma de realizar la redimensión que estaba prevista realizar. Como se dijo en entregas anteriores, esta redimensión se iba a realizar a través de QFontMetrics. Esta clase permite, entre otras cosas, medir el tamaño de un texto específico. Si se hubiese llevado a cabo este enfoque se debía calcular el tamaño y ajustar el mismo en cada creación y edición de una interfaz de tipo información. Para facilitar estos dos procesos al final se ha implementado utilizando la potencia de QML. Para ello se ha tenido que modificar parcialmente parte de la creación de las interfaces, pero este diseño permite que toda la interfaz se redimensione únicamente modificando el tamaño de la interfaz, sin necesidad de más cambios. En el apartado de Capturas de pantalla se encuentran dos capturas de la misma interfaz en la que únicamente se ha modificado el tamaño general de la ventana. Como se puede observar, el texto, automáticamente se redimensiona de forma que se ajuste a su nuevo tamaño.

La segunda implementación se centra ya en el objetivo principal de este hito, la edición. Esta implementación solamente se ha realizado para interfaces de tipo tinta. El programa accede a las Q_PROPERTY de los elementos obteniendo de esta forma los valores necesarios para poder realizar la edición. Falta corregir las situaciones excepcionales, así como realizar la parte correspondiente a las interfaces de información y menús. Se espera, dado que ya se ha comprendido la forma de proceder, que durante el próximo hito se pueda realizar estas implementaciones de forma complementaria a las planificadas para dicho hito.

Apéndice

Contenido del CD-ROM

La presente memoria dispone de un CD-ROM adjunto donde se encuentra la totalidad del proyecto. Dicho CD-ROM dispone de la siguiente estructura:

- Memoria.pdf: versión digital de la presente memoria.
- Fuentes: directorio que contiene la totalidad del código fuente desarrollado.
- Documentación: directorio que contiene la documentación del código fuente desarrollado.
- Instalador: directorio que contiene la versión de instalación de la aplicación. Al tratarse de una aplicación multiplataforma, cuenta con un instalador para sistemas Windows y otro para sistemas Unix.
- Manuales: directorio que contiene el manual de instalación, así como el manual de usuario.