

**U**niversidade

de **V**igo

**E**scola **S**uperior de **E**nxeñaría **I**nformática

|  |
| --- |
| Memoria do Traballo de Fin de Grao que presenta  D. Daniel Camba Lamas  para a obtención do Título de Graduado en Enxeñaría Informática  Editor ‘Drag&Drop’ para la generación de archivos XML con la descripción del comportamiento interactivo de un prototipado falso. |

|  |  |
| --- | --- |
|  | Xuño, 2017  Traballo de Fin de Grao Nº:  **Titor/a:** Javier Rodeiro Iglesias  Área de coñecemento:  **Departamento:** |

A ***Manuel*** y ***Pastora***, por darme los medios y el cariño para llegar hasta aquí.

A ***Diego***, ***Héctor*** y ***Román*** por las noches en vela, las de estudio y las de copas.

A ***Alba*** por nunca dejar que me rindiera y el enorme apoyo que ha sido en mi vida.

INDICE DE CONTENIDOS

[INDICE DE CONTENIDOS 4](#_Toc480744433)

[INDICE DE ILUSTRACIONES 5](#_Toc480744434)

[INDICE DE TABLAS 6](#_Toc480744435)

[INTRODUCCIÓN 7](#_Toc480744436)

[OBJETIVOS 7](#_Toc480744437)

[RESUMEN DE LA SOLUCIÓN PROPUESTA 8](#_Toc480744438)

[PLANIFICACIÓN Y SEGUIMIENTO 8](#_Toc480744439)

[ARQUITECTURA 8](#_Toc480744440)

[TECNOLOGÍAS E INTEGRACIÓN DE PRODUCTOS DE TERCEROS 9](#_Toc480744441)

[ESPECIFICACIÓN Y ANÁLISIS DE REQUISITOS 10](#_Toc480744442)

[DISEÑO DEL SOFTWARE (ESTÁTICO Y DINÁMICO) O DEL HARDWARE 10](#_Toc480744443)

[GESTIÓN DE DATOS Y INFORMACIÓN 11](#_Toc480744444)

[PRUEBAS 11](#_Toc480744445)

[MANUAL DE USUARIO 11](#_Toc480744446)

[PRINCIPALES APORTACIONES 11](#_Toc480744447)

[CONCLUSIONES 11](#_Toc480744448)

[VIAS FUTURAS DE TRABAJO 11](#_Toc480744449)

[REFERENCIAS 11](#_Toc480744450)

[APARTADO ADICIONALES 11](#_Toc480744451)

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

INTRODUCCIÓN

Si pensamos por un momento cada una de las interfaces que tenemos a nuestro alcance a diario: La aplicación que usamos para leer noticias, el sistema operativo de nuestro móvil u ordenador, la botonera del coche o nuestro microondas. Todas ellas han pasado por un proceso de diseño (más o menos) riguroso, y en las fases de ese diseño nos encontramos con el *prototipado falso*, donde el diseñador de interfaces expone a los demás integrantes del equipo o al cliente, un esbozo de cómo funcionará la aplicación, la botonera de una máquina, etc.

Dicho esbozo será un conjunto de imágenes ordenadas, donde el diseñador nos expone las diferentes vistas y/o comportamientos que derivan de la interacción con dicha interfaz. Pero pese al orden detallado que pueda darnos el diseñador, la visualización que obtenemos de la interfaz resulta estática y por ello, surge este proyecto…

Este proyecto llevará a cabo la implementación de una parte del trabajo teórico llevado a cabo por el tutor de éste proyecto **Javier Rodeiro Iglesias** y que posteriormente fue continuado por **Pedro Miguel Teixeira Faria**, donde se define pormenorizadamente lo que es una interfaz, por qué está compuesta y como estos componentes interaccionan para crear dicha interfaz.

En el marco teórico citado se define que los componentes más básicos de una interfaz pueden definirse mediante dibujo o mediante imágenes, por lo que por motivos de alcance en este proyecto nos basaremos en las imágenes.

Así el objetivo será implementar un sistema de interacción con imágenes rasterizadas (BMP, JPG, PNG) siendo PNG el formato preferido debido al soporte de transparencias, donde el usuario sea capaz de definir elementos simples y complejos de la interfaz y la interacción entre ellos de la manera más intuitiva posible.

Maquetando así el *prototipado falso*, con sus diferentes vistas y comportamientos. Para obtener finalmente un archivo XML donde estará definida toda la interfaz como la interacción entre sus componentes, organizado de manera jerárquica.

Para que posteriormente dicho archivo pueda ser cargado en un visualizador que interpretará las etiquetas del XML a fin de otorgar una experiencia dinámica e interactiva sobre el *prototipado falso*. Sin embargo, dicho visualizador está fuera del alcance de este proyecto.

OBJETIVOS

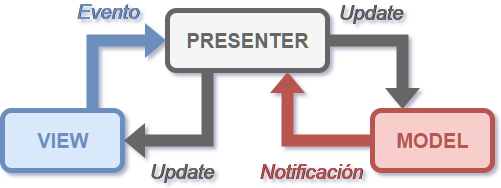
* Construir una app gráfica de escritorio programada con Qt que resulte intuitiva.
* Que permita operaciones básicas con imágenes: Mover y Escalar.
* Que acepte componentes simples, basado en imágenes, los cuales agregan la necesidad de conocer:
  + Posición
  + Tamaño
  + Nivel de profundidad
  + Si es visible en la escena
  + Si está activo en la escena
* Que permita además agregar componentes complejos, formados por componentes simples u otros componentes complejos, los cuales:
  + Permite definir estados, siendo un estado el conjunto de características de los componentes que lo componen […] !
  + Permiten definir los eventos que desencadenan las transiciones entre los estados (a nivel de alcance se han limitado estos eventos a los eventos soportados por dispositivos táctiles)
  + Permiten definir si dichas transiciones tienen alguna precondición, siendo una precondición la dependencia de que otro componente se encuentre en un estado determinado.
* Que permita el manejo de dichos componentes desde un área de trabajo.
* Que permita el manejo de dichos componentes desde fuera del área de trabajo, mostrando una tabla con las características más importantes de cada componente y permita realizar las mismas acciones que desde el área de trabajo a excepción de mover y escalar, las cuales se consideran acciones únicamente visuales.
* Que permita, en base a la especificación del marco teórico del que parte este proyecto:
  + Guardar el estado del proyecto en un archivo XML donde de manera jerárquica y con etiquetas específicas, se define con exactitud la interfaz que se está definiendo en base a componentes, los estados de estos y las interacciones requeridas.
  + Cargar un archivo XML de forma que recupere el proyecto exactamente donde lo dejamos y poder así hacer modificaciones en los componentes que ya teníamos definidos o definir componentes nuevos.
* Que permita visualizar su interfaz en varios idiomas.

RESUMEN DE LA SOLUCIÓN PROPUESTA

PLANIFICACIÓN Y SEGUIMIENTO

ARQUITECTURA

Al tratarse de una aplicación para escritorio, con interfaz gráfica, la arquitectura utilizada será una arquitectura en capas siguiendo el patrón ***MVP con vista pasiva***; el cual es una evolución del archiconocido *MVC*.



En ***MVP con vista pasiva***, la ***V****ista* sólo define la interfaz de usuario, deposita toda la lógica de interfaz en el ***P****resentador* el cual posee la *lógica de negocio* y ejecuta la *lógica de interfaz* cuando se dispara un *Evento de usuario*; lo que reduce la cantidad de código, aprovechando la conexión de datos. En base a ello, el ***P****resentador* actualiza el ***M****odelo* el cual una vez actualice la capa de persistencia, notificará al ***P****resentador* de los cambios y éste actualizará la ***V****ista.*

TECNOLOGÍAS E INTEGRACIÓN DE PRODUCTOS DE TERCEROS

Tratándose de una aplicación con interfaz gráfica, resultaba obvio la necesidad de una librería o framework que cubriera dicha necesidad. Dado que la aplicación ha sido pensada para escritorio y el rendimiento de *Electron.io* es bajo, se descartó el uso de tecnologías web; por lo que los candidatos más populares eran GTK y Qt.

El desarrollo de la aplicación quería llevarse a cabo utilizando Python y aunque ambas tenían *bindings* para el lenguaje, se ha utilizado finalmente Qt por resultar (subjetivamente) más intuitivo. A mayores de que Qt provee de más elementos multiplataforma que sólo elementos de GUI como el tratamiento de hilos y sistema de ficheros, cosa que GTK no posee. Y esto hace que de querer actualizar la aplicación con nuevas funcionalidades complejas y multiplataforma, resulte más sencillo.

Respecto a la lógica de negocio y otras características de la interfaz que no pudieron resolverse por carencias de PyQt:

* **Python-i18n[YAML] (**<https://github.com/tuvistavie/python-i18n>**)**: Para leer archivos YAML con el texto de la interfaz en diferentes idiomas (Español, Inglés, Frances y Alemán)
* **Lxml (**<http://lxml.de>**)**: Para la persistencia de datos, ya que guardaremos y cargaremos el estado de un proyecto en un archivo XML.
* **collections.deque (*interna de Python*)**: Será nuestro medio de almacenamiento en tiempo de ejecución. Resulta más eficientes que los arrays básicos.
* **copy (*interna de Python*)**: Para el funcionamiento de las acciones *Hacer/Deshacer* es necesario hacer copias exactas de objetos.
* **queue.LifoQueue (*interna de Python*)**: Para guardar los estado de hacer se usará una pila, por mayor eficiencia y reducción de posibles errores.
* **xml (*interna de Python*)**: Por una función llamada *toprettyxml* que facilita la legibilidad humana del fichero creado utilizando la librería **lxml**.

ESPECIFICACIÓN Y ANÁLISIS DE REQUISITOS

|  |  |
| --- | --- |
| REQUISITOS  FUNCIONALES | REQUISITOS  NO FUNCIONALES |
| El software debe poder *Gestionar* un proyecto. Es decir, crear, guardar estado del proyecto y carga el estado de un proyecto. | Ser de multiplataforma. |
| El software debe poder *Gestionar* componentes simples. Es decir, crear componentes simples a partir de una imagen, borrar el componente del área de trabajo y modificar distintos valores visuales. | Imitar dentro de lo posible a otras aplicaciones que trabajan con imágenes para así resultar lo más intuitiva posible, imitando acciones, atajos de teclado, ... |
| El software debe poder acceder al sistema de ficheros del usuario para cargar las imágenes. | Funcionar off-line. |
| El software debe poder realizar operaciones básicas sobre las imágenes. (Mover y Escalar) | Mostrar el último cambio realizado, zoom del área de trabajo y, otros datos de interés en la barra de estado. |
| El software debe poder *Gestionar* componentes complejos. Es decir […] | Almacenar estado cada vez que se produce un cambio para soportar las acciones *Deshacer* y *Rehacer* |
| El software debe poder *Parsear* objetos de Python a XML. |  |
| El software debe poder *Parsear* de XML a objetos Python. |  |

DISEÑO DEL SOFTWARE (ESTÁTICO Y DINÁMICO) O DEL HARDWARE

GESTIÓN DE DATOS Y INFORMACIÓN

PRUEBAS

MANUAL DE USUARIO

PRINCIPALES APORTACIONES

* Agregar Z a la definición […]

CONCLUSIONES

VIAS FUTURAS DE TRABAJO

REFERENCIAS

APARTADO ADICIONALES