

Universidad de Jaén

Escuela Politécnica Superior de Jaén

Departamento de Informática

Don Francisco Daniel Pérez Cano y Don Juan José Jiménez Delgado, tutores del Proyecto Fin de Carrera titulado: Desarrollo de un Serious Game, que presenta David Pulido Marchal, autorizan su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, MES de 2023

El alumno: Los tutores:

David Pulido Marchal NOMBRE \_TUTORES

Índice

[Índice 2](#_Toc172205449)

[1. INTRODUCCIÓN 3](#_Toc172205450)

[1.1. Motivación 3](#_Toc172205451)

[1.2. Objetivos del trabajo 4](#_Toc172205452)

[1.3. Análisis de software, estudio de alternativas y viabilidad 4](#_Toc172205453)

[1.3.1. Motor gráfico 4](#_Toc172205454)

[1.3.2. Entorno de desarrollo 6](#_Toc172205455)

[1.3.3. Git GUI client 7](#_Toc172205456)

[1.3.4. Edición de imágenes #TODO 8](#_Toc172205457)

[1.4. Tecnologías utilizadas 8](#_Toc172205458)

[1.4.1. Equipo de desarrollo 8](#_Toc172205459)

[1.4.2. Software empleado 9](#_Toc172205460)

[1.4.2.1. Unity 9](#_Toc172205461)

[1.4.2.2. Visual Studio Code 10](#_Toc172205462)

[1.4.2.3. Fork 11](#_Toc172205463)

[1.4.2.4. GIMP 11](#_Toc172205464)

[1.4.2.5. Audacity 12](#_Toc172205465)

[1.4.2.6. Probuilder 12](#_Toc172205466)

[1.5. Metodología de desarrollo 12](#_Toc172205467)

[1.6. Alcance 13](#_Toc172205468)

[2. ANÁLISIS 14](#_Toc172205469)

[1.7. Requisitos 14](#_Toc172205470)

[1.7.1. Requisitos Funcionales 15](#_Toc172205471)

[1.7.2. Requisitos No Funcionales 16](#_Toc172205472)

[1.8. Planificación 17](#_Toc172205473)

[1.8.1. Tareas #TODO 17](#_Toc172205474)

[Bibliografía 19](#_Toc172205475)

# INTRODUCCIÓN

En esta sección se intentarán explicar las distintas motivaciones y las diferentes explicaciones de la decisión de este tema y sobre el trasfondo del mismo en la sociedad actual.

## Motivación

Recuerdo mi primera experiencia con un ordenador. Por aquel entonces era normal el uso de máquinas de escribir para realizar documentos, por lo que mis padres decidieron apuntarme a clases de mecanografía. Llegaba, empezaba una lección y acababa y así una y otra vez, poco a poco iba teniendo esa soltura necesaria y cada vez hacía las cosas más rápido, pero para un niño que acababa de entrar a clase y que veía que sus amigos salían a jugar y se lo pasaban en grande pues la verdad es que era como volver a clase otra vez por la tarde. El problema no era en sí hacer una actividad extraescolar, algunos de mis amigos practicaban algún deporte y recuerdo que se lo pasaban muy bien, pero no veía ningún incentivo en esas clases. Al cabo del tiempo de la clase llena solo quedamos la mitad de esta y los pocos que seguíamos íbamos más por el hábito y por el tiempo que ya habíamos invertido más que por el propio placer que suponían las clases o el hecho de la mejora de la habilidad en la mecanografía.

Para sorpresa de mucha gente, a día de hoy con todos los ordenadores instaurados en el día a día y con el auge de los dispositivos inteligentes, solo el 20% de las personas saben escribir correctamente con un teclado mientras que el 80% restante suele escribir mirando el teclado lo que reduce la eficiencia del uso del mismo y disminuye la velocidad a la hora de escribir en un ordenador[B].

Prácticamente todos los empleos y niveles de educación requieren o se van adaptando para requerir el uso de sistemas informáticos para un desempeño eficiente. Todas las empresas están compitiendo por ser las primeras en lograr una transformación digital, ya que reconocen que esta evolución tecnológica es un factor determinante en su supervivencia y éxito en un mundo empresarial cada vez más dinámico y digitalizado, por lo que lo van instaurando poco a poco en cada una de sus partes y esta tendencia está también instaurándose en el sector educativo. Asimismo, el 30% de las empresas no contratarían a alguien que no supiera escribir con un teclado y que además lo consideran una habilidad fundamental en la vida[A]. En el ámbito educativo, un dato relevante es que el 76 % de los niños británicos de edades comprendidas entre los 7 y los 13 años emplea una computadora para realizar sus deberes escolares, mientras que un 24 % no lo hace. [A].

Por lo que hemos visto, la mecanografía es una ciencia necesaria para el uso correcto y eficiente de las TIC (Tecnologías de la Información y la Comunicación) ya que, aunque se pueda usar correctamente los distintos programas o las herramientas que nos proporciona, una gran parte del mismo pasan por escribir en el ordenador los distintos comandos o por comunicarse a través de distintos medios como correos electrónicos, redes sociales o herramientas de colaboración como es Microsoft Teams.

## Objetivos del trabajo

El objetivo general es el desarrollo de un prototipo de un videojuego para el aprendizaje de mecanografía usando la plataforma de desarrollo en tiempo real de Unity, ofreciendo distintos niveles que guíen al usuario a la hora de aprender poco a poco todas las teclas del teclado y que planten una serie de desafíos que prueben la velocidad y la precisión a la hora de teclear.

## Análisis de software, estudio de alternativas y viabilidad

Antes de empezar a desarrollar el desarrollo del videojuego, como en cualquier proyecto de desarrollo se tiene que evaluar las diferentes tecnologías existentes en el mercado y enumerar las distintas diferencias a nivel económico, documentación, porcentaje de uso.

### Motor gráfico

Tras hacer una investigación y según una encuesta realizada por el DEV 2022, los dos motores más usados son Unity y Unreal Engine en la industria del videojuego en España[C]

A continuación, se van a presentar los diferentes motores gráficos que se han estudiado y las diferencias entre los mismos:

|  |  |  |
| --- | --- | --- |
|  | Unity 3D | Unreal Engine 5 |
| Fecha de lanzamiento inicial | 8 de junio de 2005 | 5 de abril de 2022 |
| Lenguaje de programación | C# | C++ |
| Plataformas compatibles | Windows (PC), Mac, Universal Windows Platform (UWP), Linux Standalone, iOS, Android, ARKit, ARCore, Microsoft HoloLens, Windows Mixed Reality, Magic Leap (Lumin), Oculus, PlayStation VR, PS5, PS4, Xbox One, Xbox X|S, Nintendo Switch, Google Stadia, WebGL  Embedded Linux, QNX | Windows PC, PlayStation 5, PlayStation 4, Xbox Series X, Xbox Series S, Xbox One, Nintendo Switch, macOS, iOS, Android, ARKit, ARCore, OpenXR, SteamVR, Oculus, Linux, and SteamDeck |
| Desarrollado por | Unity Technologies | Epic Games |
| IDE compatible | Windows, Mac, Linux | Windows, Mac, Linux |
| Características destacables | Mejoras a los entornos 2D, animaciones, creación de snapshots | Un framework robusto para juegos multijugador, VFX y un simulador de particulas |
| Código fuente | Not open-source. | No considerado open-source, pero se puede acceder al código del motor |
| Pricing | Gratis | Gratis |
| Facilidad de Uso | Curva de aprendizaje sencilla | Dificil de aprender |
| Gráficos | Buenos gráficos generales, pero menos refinados que Unreal | Gráficos fotorrealistas utilizados en juegos AAA |
| Otros aspectos | Tiene una comunidad grande y activa, lo que facilita la colaboración y la resolución de problemas, además de tener mucha documentación |  |

Tabla 1.1 Características de los motores gráficos

* Similitudes:
  + Gratuitos
  + Ambos no son open-source
  + Permiten lanzar el videojuego desarrollado en distintas plataformas, incluidas las más modernas del mercado, así como entornos de realidad virtual
* Diferencias
  + La curva de aprendizaje de Unity es menor
  + En Unity hay un mayor número de tutoriales y de guías. La documentación de Unreal está peor organizada y es más liosa de entender
  + Unity es más usado dentro de la comunidad española

Con respecto al precio, ambos motores se cobran un porcentaje de regalías por ingresos brutos en ventas de productos comerciales una vez que se alcanza un umbral determinado, pero como en este caso concreto se usa como un recurso educativo o de investigación en lugar de ser un producto comercial seguirá siendo gratuito, pero esto hay que tenerlo en cuenta si se quiere calcular los costes para lanzarlo como un producto comercial.

### Entorno de desarrollo

|  |  |  |
| --- | --- | --- |
|  | Visual Studio Code | Visual Studio |
| Fecha de lanzamiento inicial | 29 de abril de 2015 | 1 de mayo de 1997 |
| Desarrollado por | Microsoft | Microsoft |
| Programado en | TypeScript, JavaScript, CSS | C++ y C# |
| Entorno de desarrollo | No, pero posible a través de extensiones | Si |
| Plataformas | x86, x86-64 y ARM | x86-64 |
| Lenguajes soportados | Multiples (C++, C#, Fortran, .NET, Java, Python, PHP,...) | Multiples (C++, C#, Fortran, .NET, Java, Python, PHP,...) |
| Código fuente | Técnicamente open-source pero su descarga está privatizada por Microsoft | No considerado open-source, pero se puede acceder al código del motor |
| Precio | Gratis | Gratuito con opciones de pago |
| Tamaño | Ligero (200MB) | Pesado (min 2,3GB y 16GB RAM) |
| Otros aspectos | Altamente personalizable aunque depende mucho de los plugin de la comunidad | Fuerte entorno de pruebas e integración con Azure. La instalación es sencilla e incluye todos los elementos necesarios para el desarrollo. Puede ser más complejo de lo necesario |

Tabla 1.2 Características de los entornos de desarrollo

* Similitudes:
  + Gratuitos
  + Ambos pertenecen a Microsoft
  + Pueden ser ejecutados en ordenadores modernos cuyos procesadores funcionen mediante CISC (Complex Instruction Set Computer)
  + Ambos permiten gran cantidad de lenguajes
* Diferencias
  + Visual Studio Code es más ligero que Visual Studio, lo cual permite ejecutarse en ordenadores que tengan menos recursos
  + Visual Studio Code permite ejecutarse en arquitecturas ARM
  + Visual Studio Code es un editor de código que debe gran parte de su soporte a la comunidad que desarrolla las extensiones mientras que Visual Studio tiene un equipo dedicado a desarrollar sus complementos que están bien incorporados

### Git GUI client

Git es un sistema de control de versiones distribuido que rastrea las versiones de los archivos. Los programadores que desarrollan software de forma colaborativa suelen utilizarlo para controlar el código fuente. [M] Git es un software gratuito y de código abierto compartido bajo la licencia GPL-2.0 únicamente.

|  |  |  |
| --- | --- | --- |
|  | SmartGit | Fork |
| Fecha de lanzamiento inicial | 10 de agosto de 2009 | 17 de abril de 2016 |
| Desarrollado por | syntevo GmbH | Dan Pristupov y Tanya Pristupova |
| Programado en | Java[O] | C#+WPF para Windows y Swift+Cocoa en Mac[N] |
| Sistemas Operativos | Windows, Mac y Linux | Windows and Mac |
| Precio | Gratuito con licencia no comercial ligado a una institución de investigación. En caso comercial, de pago como mínimo 89€. También ofrece modelo de subscripción | 60$ con una evaluación gratuita de uso indefinido |
| Otros Aspectos | Posee integraciones con otras aplicaciones y servicios como AzureDevOps, Jenkins, JIRA...[P] |  |

Tabla 1.3 Características de los clientes de Git

* Similitudes:
  + Ambos son productos de pago que poseen una licencia gratuita
  + Pueden ser instalados en Windows
  + Poseen todas las funcionalidades básicas de Git, así como integración con Github
* Diferencias
  + SmartGit posee una licencia más cara y además limita las actualizaciones [Q] mientras que Fork posee una licencia de pago único con acceso hasta 3 dispositivos
  + SmartGit tiene más funcionalidades e integraciones

### Edición de imágenes #TODO

|  |  |  |
| --- | --- | --- |
|  | GIMP | Photoshop |
| Fecha de lanzamiento inicial | 15 de febrero de 1996 | 17 de abril de 2016 |
| Desarrollado por | syntevo GmbH | Dan Pristupov y Tanya Pristupova |
| Programado en | C y GTK | C#+WPF para Windows y Swift+Cocoa en Mac[N] |
| Sistemas Operativos | Windows, Mac y Linux | Windows and Mac |
| Precio | Gratuito con licencia no comercial ligado a una institución de investigación. En caso comercial, de pago como mínimo 89€. Tambien ofrece modelo de subscripción | 60$ con una evaluación gratuita de uso indefinido |
| Otros Aspectos | Posee integraciones con otras aplicaciones y servicios como AzureDevOps, Jenkins, JIRA...[P] |  |

Tabla 1.4 Características de los editores de imágenes

## Tecnologías utilizadas

### Equipo de desarrollo

|  |  |  |
| --- | --- | --- |
| Equipo | Asus VivoBook S512DA-BR097 | Tower |
| CPU | Ryzen R5 3500U: 4 Núcleos, 2MB Caché, 2.0GHz hasta 3.6GHz, 64-bit | Intel(R) Core (TM) i5-9600K CPU @ 3.70GHz 3.70 GHz |
| RAM | 8 GB | 16GB |
| Disco duro | *500 GB SSD* | 256GB SSD  1TB HDD |
| Resolución de pantalla | 1366x768 | 3440 x 1440 |

Tabla 1.5 Especificaciones de los equipos informáticos

Los elementos en cursiva dentro de la tabla han sido marcados así puesto que no iban incluidos originalmente en los equipos, sino que han sido añadidos o han sustituido a otros componentes que el equipo correspondiente llevaba originalmente.

### Software empleado

### Unity

El motor gráfico núcleo central de la funcionalidad de cualquier videojuego. Sin él un renderizado de gráficos 2D o 3D, un motor de físicas, un detector de colisiones y sus respuestas, el sonido, el ensamblado de scripts y su definición de los comportamientos relacionados con los distintos objetos, la generación de sombras, el uso de animaciones, el uso de inteligencia artificial, juego o funciones online no serían posibles.

Normalmente los motores gráficos proveen una serie de herramientas de desarrollo visual que están provistos por un entorno de desarrollo para poder acceder de forma rápida y sencilla a estas. Muchos desarrolladores intentan anticiparse a las necesidades e implementan paquetes de software con funcionalidad como IA, física, gráficos o sonido. Estos paquetes normalmente se conocen como middleware porque como en el sentido comercial del término, proporcionan una plataforma de software flexible y reutilizable con una funcionalidad básica necesaria lista para usar lo cual consigue reducir costes y tiempo de desarrollo[D]. Los motores de videojuegos suelen proveer de abstracción de plataforma, lo que permite que el mismo juego se ejecute en varias plataformas (incluidas consolas de juegos y computadoras personales) con pocos cambios, si es que se realizan alguno, en el código fuente del juego. Algunos de los componentes del motor se pueden quedar obsoletos por lo que se suelen dedicar algunos desarrolladores a actualizar ya sea el componente obsoleto como modificar el motor para desarrollar la funcionalidad existente.

Algunas veces estos cambios se van agrupando con el tiempo creando un árbol de desarrollo como pasó por ejemplo con el motor Quake de id Software (Cita Imagen)

Unity es un motor de videojuego que utiliza como lenguaje de programación C# que con su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes[G], como gráficos puede usar OpenGL, Direct3D o OpenGL ES, además Unity es conocido por su facilidad de uso, versatilidad y amplia gama de características y herramientas que facilitan la creación de contenido interactivo de alta calidad. En principio tiene 4 sistemas de suscripción hasta la fechaDiagrama

Descripción generada automáticamente con distintos precios y funcionalidades [H]:

Ilustración 1.1 Árbol de motores de la familia id Tech

Luego de contrastar los distintos rasgos de los motores se ha decidido usar **Unity Engine**. Lo primero de todo, este motor se ha usado anteriormente en la carrera en la asignatura de Desarrollo de Videojuegos por lo que se dispone más experiencia y por tanto no hay que pasar por un proceso de aprendizaje inicial. Seguidamente, el hecho de tener mucha documentación y una comunidad amplia y activa hace que el proceso de aprendizaje y resolución de problemas sea más accesible y eficiente lo cual reducirá el tiempo de desarrollo y por tanto sea más fácil. Finalmente, el hecho de ser el motor más usado para las distintas empresas de videojuegos de España hace probable que se tenga más documentación en español, lo cual puede suponer una ventaja si se quiere proponer este producto a alguna empresa española para poder continuar el desarrollo

### Visual Studio Code

Para toda la parte de la programación se planteó usar un IDE que se había usado anteriormente para desarrollar videojuegos con Unity, Visual Studio 2019. Sin embargo, aunque dispone de muchas facilidades y perfecta compatibilidad con Unity, a su vez es un programa muy pesado y por tanto ralentizaba el proceso de programación, haciendo de todo el proceso de desarrollo muy ineficiente. Por lo que se planteó una alternativa: Visual Studio Code.

Visual Studio Code es un editor de código fuente que no un entorno de desarrollo integrado desarrollado por Microsoft para Windows, Linux, macOS y Web gratuito y de código abierto basado en Electron, un framework que se utiliza para implementar Chromium y Node.js como aplicaciones para escritorio. Es bastante más ligero que Visual Studio y permite la instalación de una amplia variedad de extensiones disponibles para agregar funcionalidad adicional según las necesidades. Visual Studio Code es conocido por ser rápido y eficiente en términos de recursos.

Por tanto, para este proyecto viendo las características del equipo que se ha estado utilizando y por la existencia de dos distintas extensiones que son Unity for Visual Studio Code y C# Dev Kit se ha decantado por este editor de código fuente.

### Fork

Aunque Visual Studio Code tiene una extensión que permite el uso de ramas dentro del mismo se prefiere un entorno visual para la gestión de las distintas ramas que se emplean en el proyecto. Incialmente en la carrera se usó SmartGit pero había un par de problemas. El primero fue la licencia de uso, si bien actualmente se posee una licencia de uso no comercial gracias a ser estudiante, tras entregar el proyecto dejariamos de tener acceso a esta licencia y además si luego se intenta comercializar este prototipo estaríamos incumpliendo la misma.

Por lo tanto se buscó una herramienta fácil, moderna que posea todos los requerimientos típicos, que sea altamente actualizado. Asique este programa que si bien tiene un coste de 59,99$ se puede usar perfectamente con su evaluación gratuita indefinida.

### GIMP

GIMP es un editor de gráficos rasterizados gratuito y de código abierto que se utiliza para la manipulación o retoques y edición de imágenes, dibujo de forma libre, transcodificación entre diferentes formatos de archivos de imagen y tareas más especializadas. Es extensible mediante complementos y se puede programar. No está diseñado para usarse para dibujar, aunque algunos artistas y creadores lo han utilizado de esta manera. Es multiplataforma, lo que significa que puede ejecutarse en varios sistemas operativos, incluyendo Windows, macOS y Linux.

La alternativa a este software era usar Adobe Photoshop, un software propietario desarrollado y publicado por Adobe Inc. Es un editor de fotografías, usado principalmente para el retoque de fotografías y gráficos. Es ampliamente considerado como el estándar de la industria en cuanto a edición de imágenes y gráficos, y es utilizado por profesionales en diversos campos, incluyendo diseño gráfico, fotografía, web, y producción de vídeo. Photoshop puede editar y componer imágenes rasterizadas y soporta varios modelos de colores como RGB, CIELAB, CMYK y para ello ha desarrollado un formato propio como PSD y PSB.

Los principales motivos para escoger GIMP son que es un software gratuito y aunque puede carecer de algunas de las funcionalidades más avanzadas presentes en Photoshop como el uso de IA Generativa, tiene las funcionalidades necesarias para desarrollar un prototipo y una comunidad suficientemente grande como para desarrollar extensiones y plugins y realizar tutoriales que permiten desarrollar los assets necesarios en un videojuego.

### Audacity

Audacity es una aplicación informática multiplataforma libre que se puede usar para grabación y edición de audio, Se ha utilizado para la normalizar, editar y recortar todos los audios usados en el videojuego.

### Probuilder

Probuilder es un plugin desarrollado para Unity, es un híbrido entre una herramienta de modelado 3D y un diseñador de niveles. Los objetos 3D usados en el videojuego han sido modelados a través de esta herramienta.

## Metodología de desarrollo

El modelo incremental es un enfoque de desarrollo de software donde el producto se diseña, implementa y prueba en incrementos pequeños y manejables. En lugar de entregar el producto en su totalidad este se divide en incrementos de tal modo que cada entrega es un producto operativo pero incompleto cuyo periodo de tiempo es breve. Los incrementos se estructuran de forma que los requisitos principales se desarrollan en los primeros incrementos. Cada incremento pasa por un ciclo completo de desarrollo, que incluye el análisis, diseño, implementación y pruebas. Los requisitos del incremento no suelen ser modificados hasta que ha sido terminado, aunque puede ser común surgir cambios de un incremento a otro, se suelen incluir en la planificación de un incremento posterior. [K][L]

Este proceso de desarrollo de software tiene una serie de ventajas que podemos definir como:

* Los errores de los requisitos importantes son reconocidos en etapas tempranas de desarrollo.
* Los clientes pueden usar el software que poseen las características más importantes en etapas tempranas y por tanto dar feedback relevante
* El riesgo de que el proyecto no llegue a una etapa final disminuye
* Los incrementos primordiales poseen un mayor número de pruebas.
* La carga de trabajos es más manejable y permite distribuir más fácilmente el trabajo
* La gran cantidad de entregas permite que el cliente se mantenga involucrado con el proyecto

Y las principales desventajas son las siguientes:

* Requiere de una gran capacidad de planificación
* Dificultad a la hora de acomodar determinados requisitos en un incremento

## Alcance

Una vez definidos los objetivos del proyecto, se debe determinar qué documentos se incluirán:

* **Ejecutable operativo**: Es un videojuego compilado que debe ser entregado en la fecha fijada y cumplir con todos los requisitos listados. Será accesible a través de internet descargándolo a través del repositorio de github o a traves del ejecutable subido a la plataforma.
* **Código fuente**: En la entrega final se incluirá el código fuente necesario, no solo para arrancar el videojuego, sino también para su consulta, dado que se hace referencia a este a lo largo del documento. La etapa de desarrollo se explica detalladamente en el punto X.TODO
* **Manual de instalación**: Al final del documento se incluye una sección que explica cómo descargar y ejecutar el entorno de desarrollo del videojuego. (TODO ANEXO)
* **Manual de usuario**: Al final del documento existe otro apartado que detalla los controles y funcionalidades de la interfaz de cada escena. (TODO ANEXO)

# ANÁLISIS

## Requisitos

En ingeniería, un requisito es una condición que debe cumplirse para que el resultado de un trabajo sea aceptable. Es una descripción explícita, objetiva, clara y a menudo cuantitativa de una condición que debe cumplir un material, diseño, producto o servicio. [R]

El Glosario estándar IEEE de terminología de ingeniería de software define un requisito como[S]:

1. Una condición o capacidad que necesita un usuario para resolver un problema o alcanzar un objetivo
2. Una condición o capacidad que debe cumplir o poseer un sistema o un componente del sistema para satisfacer un contrato, una norma, una especificación u otro documento impuesto formalmente
3. Una representación documentada de una condición o capacidad como en 1 o 2

Los requisitos en ingeniería de software suelen clasificarse en dos categorías:

* **Requisitos funcionales**: Describen lo que hace un sistema o lo que se espera que haga, es decir, su funcionalidad. Suelen ir acompañados de diagramas de casos de uso. Una función es descrita como un conjunto de entradas, comportamientos y salidas.
* **Requisitos no funcionales**: describen aspectos del sistema que están relacionados con el grado de cumplimiento de los requisitos funcionales. Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento. Suelen ser las restricciones o condiciones que impone el cliente al programa que necesita.

### Requisitos Funcionales

1. **Interfaz de Usuario**: Conjunto de elementos que permiten al jugador seleccionar distintas opciones y que informa al jugador de los distintos datos que posee el videojuego
   1. Menú principal: El prototipo dispondrá de una pantalla de inicio que que permitirá al jugador seleccionar entre los distintos modos de juego y configurar las opciones
   2. IU: Configuración: Se dispondrá de un menú accesible desde el menú principal y en cualquier momento en cualquier modo de juego que permite configurar el sonido del juego
   3. IU: Pantalla de juego: En cualquier modo de juego se dispondrá de una interfaz de usuario clara que muestre al menos la puntuación y cualquier sistema que implique una condición de derrota
   4. IU: Modos de Juego: Se dispondrá de un menú propio para configurar algún parámetro necesario en el modo de juego
2. **Mecánicas comunes**: Conjunto de funcionalidades principales que serán comunes entre los distintos modos de juego.
   1. Generación de letras: Los elementos a destruir, conocidos como “letras”, deberán de salir en la parte superior de la pantalla pudiendo configurar su frecuencia. Además se debe de ser capaz de configurar para que su generación sea determinada.
   2. Eliminación de letras: Las “letras” deberán de ser eliminadas al pulsar una tecla concreta mientras mantengan contacto con una zona concreta del escenario conocida como “hit zone”.
   3. Número de fallos: Existe un sistema que cuenta el número de “letras” que no se ha logrado eliminar y que al alcanzar un cierto umbral indica el fin del juego
   4. Movimiento de las letras: Las “letras” deben de bajar con una velocidad configurable
3. **Puntuación**: Cualquier acción del jugador en los distintos modos de juego deben de ser reflejados en la puntuación
   1. Acierto y combos: Cuando se elimina una “letra” se deberá de aumentar la puntuación del juego. En caso de encadenar varios aciertos este aumento será mayor.
   2. Errores: En caso de que una “letra” no sea eliminada, se deberá de restar un número determinado a la puntuación del juego. En caso de que se pulse una tecla incorrecta también se reducirá la puntuación del juego.
   3. Leaderboard: Se debe de disponer de un registro de puntuaciones ordenado indicando el nombre o apodo del jugador, así como la puntuación obtenida en un modo de juego concreto con unos parámetros concretos.
4. **Modo de juego**: Random: En este modo de juego se deberán de generar las “letras” de forma aleatoria o siguiendo una distribuciones porcentuales predefinida
   1. Alfabetos: El número de alfabetos que serán posibles para escoger será el español e inglés
   2. Modos de dificultad: Se podrá escoger entre 3 modos de dificultad que modificarán la velocidad y el tiempo de generación de las “letras”.
5. **Modo de juego**: Niveles: Existirá un modo de juego en el cual se podrá seleccionar un nivel concreto que será pre-configurada y generará las “letras de forma” determinada.
   1. Selección de niveles y mundo: Los niveles estarán divididos en mundos a través de un menú

### Requisitos No Funcionales

Se proponen los siguientes requisitos no funcionales:

* El prototipo deberá de tener un ejecutable que pueda ser ejecutado en Windows
* El prototipo deberá de tener una interfaz adaptada a 16:9 sin importar la resolución
* El juego debe de proporcionar feedback inmediato sobre las acciones del usuario ya sea de forma visual o de forma auditiva
* Se deberá de disponer de una guía de controles indicando las diferentes opciones
* Todas las funciones del código deberán de estar documentadas
* La interfaz deberá de tener un código de colores que sea compatible con los distintos tipos de daltonismo
* El sistema de puntuaciones deberá de ser persistente entre sesiones
* El sistema deberá de ejecutarse

## Planificación

### Tareas #TODO

Definidos los requisitos funcionales se generan tareas para lograr llevar a cabo dicho objetivo, realizando varias subtareas:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Tarea | Subtareas | Procedentes | Duración (días) |
| A | Estudio Previo | 1. Análisis del problema 2. Estudio de metodología y tecnologías | - | 6 |
| B | Planificación | 1. Obtención de requisitos 2. Planificación de tareas 3. Estimación de costes 4. Diseño de diagramas e interfaz | A | 10 |
| C | Loop jugable | 1. Crear escena principal con elementos estáticos 2. Añadir todas las “letras” 3. Crear script de funcionalidad de bajada y eliminación | B | 2 |
| D | Definir sistema generación | 1. Definir función de generación 2. Guardar datos en un archivo csv | C | 2 |
| E | Añadir música y sonido | 1. Investigar como generar audio en Unity 2. Implementar música de fondo 3. Implementar sonidos | C | 2 |
| F | Limpieza de elementos | 1. Definir un colider para eliminar las letras no pulsadas | C | 0,5 |
| G | Alfabetos | 1. Leer e investigar estudios sobre el porcentaje de usos de las letras en inglés y español 2. Implementar uso de letras por porcentaje | - | 1 |
| H | Salud | 1. Implementar sistema de salud 2. Implementar barra visual de vidas | C | 1 |
| I | Loop jugable II | 1. Realizar modificaciones visuales del escenario 2. Implementar feedback visual 3. Añadir pausa al nivel | C | 1 |
| J | Score | 1. Añadir funciones de Score y combo 2. Visualizar Score y Combo en UI | I | 1 |
| K | UI | 1. Diseñar la interfaz en todas las pantallas 2. Añadir iconos 3. Implementar un menú principal que permita acceder al loop jugable 4. Implementar un menú de pausa 5. Implementar un slider para cada tipo de volumen. 6. Corregir errores sonido | E, I | 4 |
| L | Loop jugable III | 1. Mejorado sistema de colisiones 2. Implementado fin de juego 3. Sistema de dificultad 4. Velocidad y tiempo de generación en csv | D, I | 2 |
| M | Leaderboard | 1. Añadido sistema de usuarios 2. Definido e implemntado leaderboard 3. Leaderboard por modo de juego 4. Modificado UI de fin de juego para tener leaderboard | L. K | 4 |
| N | UI II | 1. Añadido Menú de Modo Random 2. Feedback visual los botones y letras | K, M | 1,5 |
| O | Letras 3D | 1. Diseño 3D de letras 2. Modificar escena para más efecto inmersivo | L | 0,5 |
| P | UI III | 1. Implementar menú de selección de niveles 2. Modificar iconos 3. Mejoras en interfaz del loop jugable 4. Implementar paleta de colores | N | 3 |
| Q | Niveles | 1. 5 nivel de prueba 2. Función de cálculo de tiempo 3. Modo Debug para creación de niveles | C | 5 |
| R | Modo Random | 1. Implementar diccionario inglés 2. Corregir errores de generación | G, L | 1 |

Tabla 2.1 Descripción de las tareas totales

Incrementos:

|  |  |  |  |
| --- | --- | --- | --- |
| INCREMENTO | INICIO | FIN | TAREAS |
| 0 |  |  | A, B |
| 1 |  |  | C |
| 2 |  |  | D, E, F, G |
| 3 |  |  | H, I, J, K |
| 4 |  |  | L, M, N, O |
| 5 |  |  | P, Q, R |

Tabla 2.2 Especificaciones de incrementos

### Diagrama de Gantt #TODO

#TODO introducir diagrama en png y hacer una carpeta en drive

El anterior diagrama se muestran todas las tareas de forma secuencial por la realización de este proyecto por una sola persona.

### Variaciones

## Estimación de costes #TODO las X

En cualquier proyecto y sobre todo en un prototipo es importante realizar una propuesta en función de las necesidades del proyecto de cara a un posible inversor. En esta propuesta se suele introducir todos los costes asociados, desde los más obvios como el salario de los trabajadores por puesto, la compra o alquiler de los equipos informáticos, el alquiler de un local y otros costes.

Para la realización del prototipo se ha propuesto un equipo de dos personas o una persona que realiza dos puestos:

* Un programador
* Un analista

Basándose en la planificación anterior el analista tiene una carga de trabajo de Xh. Según Indeed, el sueldo promedio de este puesto es de 16€/h y por tanto el coste total es de X€

Por otra parte, el programador ha trabajado Xh siendo el precio de 10€/h según Indeed por lo que el coste total es de X€

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Puesto Trabajador | Coste mensual  (5x5) | Coste (€/h) | Tiempo trabajado | Coste Total |
| Analista | 1600€ | 16€ | X | 1200€ |
| Programador | 1000€ | 10€ | X | 3200€ |
| Total |  |  |  | 4400€ |

Tabla 2.3 Salario de los trabajadores del proyecto

Como bien se ha descrito en el apartado X, gracias a que el software ha sido programas gratuitos o bien open-source no hay costes en este apartado

Un equipo informático promedio suele tener un valor de 800€ y se estima que puede llegar a 5 años de vida útil. Por tanto, los costes derivados son los siguientes.

|  |  |  |  |
| --- | --- | --- | --- |
| Valor hardware | Vida útil (años) | Amortización mensual | Amortización en X meses |
| 800€ | 5 | 13.33€ | X |
| Total |  |  | X |

Tabla 2.4 Tabla de amortizaciones

Finalmente, el presupuesto total es la suma de los costes presentados anteriormente.

|  |  |
| --- | --- |
| Descripción del coste | Valor (€) |
| Coste de personal | X |
| Coste hardware |  |
| Subtotal |  |
| Costes Indirectos (5%) |  |
| Margen (10%) |  |
| Beneficio (10%) |  |
| Antes de impuestos |  |
| IVA (21%) |  |
| Presupuesto total |  |

Tabla 2.5 Tabla de costes totales

Hay 3 puntos que se le han sumado al coste de desarrollo:

* Los costes indirectos, que suman el alquiler del espacio de trabajo, luz e internet.
* Un margen que garantiza que, en caso de cualquier imprevisto de desarrollo o un fallo de planificación, se pueda asumir el coste
* Un margen de beneficio

Calculados todos los gastos hay que sumarle los impuestos, como mínimo el IVA.

# DISEÑO

En esta sección se procede a explicar cómo está estructurado el prototipo, presentando primero las posibles acciones en cada escena o menú a través de los diagramas de caso de uso, luego se describe la interfaz indicando los distintos aspectos de diseño. Finalmente se establece un storyboard con el flujo principal.

## Diagramas de casos de uso

Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su iteración con otros usuarios o sistemas.  
El usuario tiene dos principales acciones:

* Interactuar con el sistema o interfaz
* Cambiar de escena

Por tanto, al haber en el videojuego solamente dos escenas podemos establecer dos grandes casos de uso. Menú principal y pantalla de juego.

Para una correcta comprensión de este diagrama hay varios subsistemas que serán explicados en diagramas separados, se indicarán de otro color mediante una relación de extensión.

### Menú principal

En esta escena solo se puede realizar varias opciones que son similares, abrir varios submenús:

* Abrir varios menús de selección de modo de juego:
* Modo random
* Modo niveles
* Abrir el menú de configuración que actualmente solo modifica los volúmenes de audio
* Introducir el nombre de usuario, que permite usarlo que permite usarlo en las puntuaciones y que guarda los ajustes de sonido
* Salir del juego

Diagrama

Descripción generada automáticamente

### Menú random

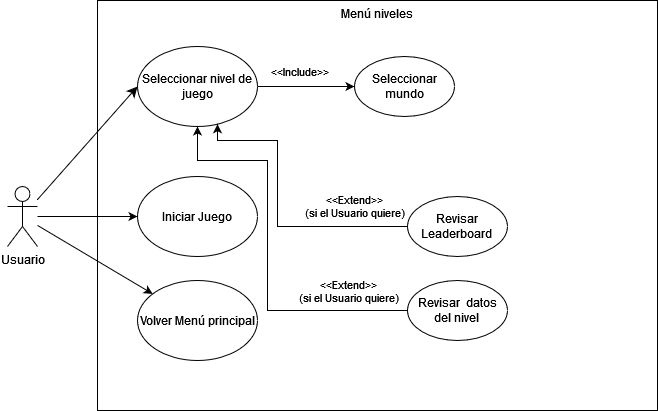
El flujo del caso de uso consiste en elegir entre los parámetros que existen y finalmente iniciar el juego. Los diferentes parámetros que actualizar son los siguientes:

* Dificultad para elegir entre 3 opciones (Fácil, Medio y Dificil)
* Idioma si se quiere usar los porcentajes de uso de inglés o español o se quiere un modo completamente aleatorio

Diagrama

Descripción generada automáticamente

### Menú niveles

El flujo de trabajo es parecido al anterior, la diferencia radica en que los parámetros ahora se distribuyen en modo árbol:

* Seleccionar Mundo
  + Seleccionar Nivel

### Menú opciones

En este menú se puede modificar los valores de sonido de tres elementos:

* General
  + Música
  + Diagrama

    Descripción generada automáticamenteEfectos Especiales

### Pantalla de juego

Esta escena es bastante compleja, aunque las acciones que puede tomar el usuario son reducidas:

* Pulsar una tecla del teclado
* Visualizar la vida
* Visualizar la puntuación y los combos
* Pausar el juego

Aunque dos opciones, Pausar el juego y Pulsar una tecla del teclado tienen más complejidad. El primero es una alteración del menú de opciones y el segundo es complejo por los diferentes estados en los cuales se puede encontrar el programa

* Sin letras en la zona de golpeo
* Con letras en la zona de golpeo y se pulsa la tecla correcta
* Con letras en la zona de golpeo y se pulsa una tecla incorrecta

De forma que los únicos estados que merecen

Diagrama

Descripción generada automáticamente

## Story Board

A continuación, se visualizan los storyboards de los casos de uso y de la aplicación.

### Texto, Carta, Pizarra Descripción generada automáticamenteMenú principal #TODO Cambiar la imagen por la interfaz actual, hay un par de cambios

Aquí se muestra el menú que aparece nada más ejecutar el juego. Podemos distinguir que las opciones principales de juego aparecen listadas primero mientras que al final están las opciones menos relevantes.

A la hora de seleccionar los juegos hay dos opciones principales, Modo Tutorial (1) y Modo Random(2). A la derecha de estos se encuentra un campo de texto en el cual se introduce el nombre del jugador (3) con su botón de guardar cambios(4).

Finalmente, en la parte inferior izquierda encontramos las dos últimas opciones, la pantalla de Ajustes y el botón de Salir. Al clicar en cualquiera de los tres menús anteriores se abre un submenú que veremos a continuación.

### Menú Niveles #TODO Cambiar la imagen por la interfaz actual, hay un par de cambios

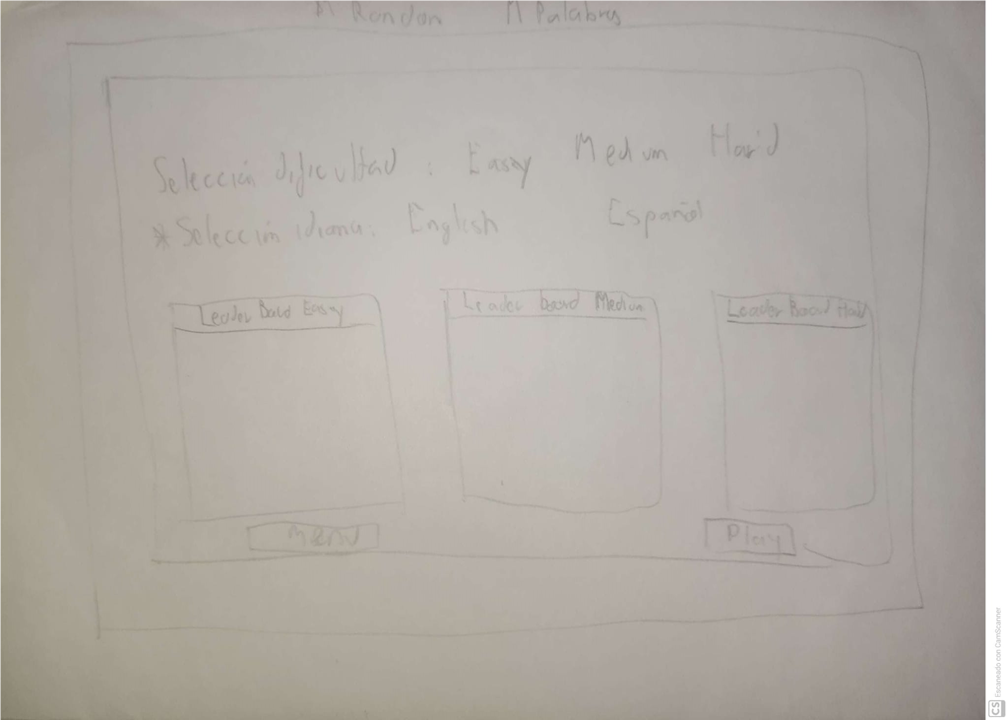
Este submenú contiene 2 apartados principales la parte superior (1) indica un selector de mundo que podrá ser cambiado mediante las teclas Q y E para avanzar y retroceder respectivamente.

Cuando se haya seleccionado el mundo se pasará a la parte central, el selector de nivel (2). Aquí a través de las letras A y D se podrá seleccionar un nivel como se ha descrito anteriormente.

Cuando se haya seleccionado el nivel veremos que en la parte central izquierda (3) se mostrará información relativa al nivel tal como la duración o la puntuación obtenida anteriormente. También se indicará en la parte central derecha (4) una tabla de puntuación con los mejores resultados anteriormente.

Para concluir, tenemos la parte inferior que tiene un botón de volver al menú principal (5) o el botón más importante de este submenú, que es el botón de iniciar el juego (6)

### Menú Random #TODO Cambiar la imagen

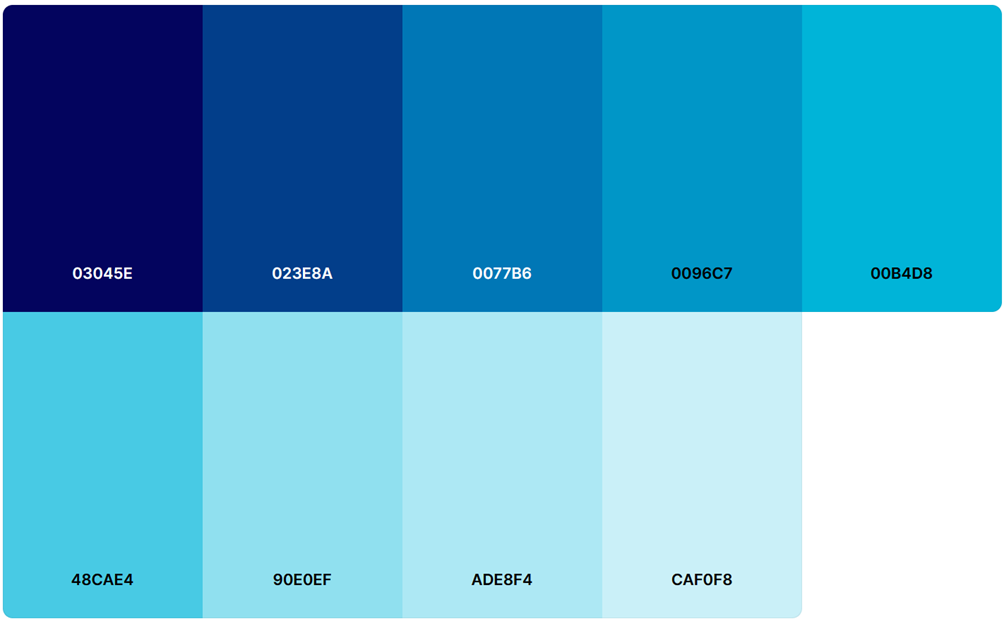
Este submenú se compone de dos partes: la parte superior (1) que posee un par de selectores indicando la dificultad del juego y si se quiere seleccionar un idioma a la hora de generar las “letras” y la parte central (2) que muestra diferentes tablas de puntuación con los mejores resultados agrupados por nivel de dificultad.

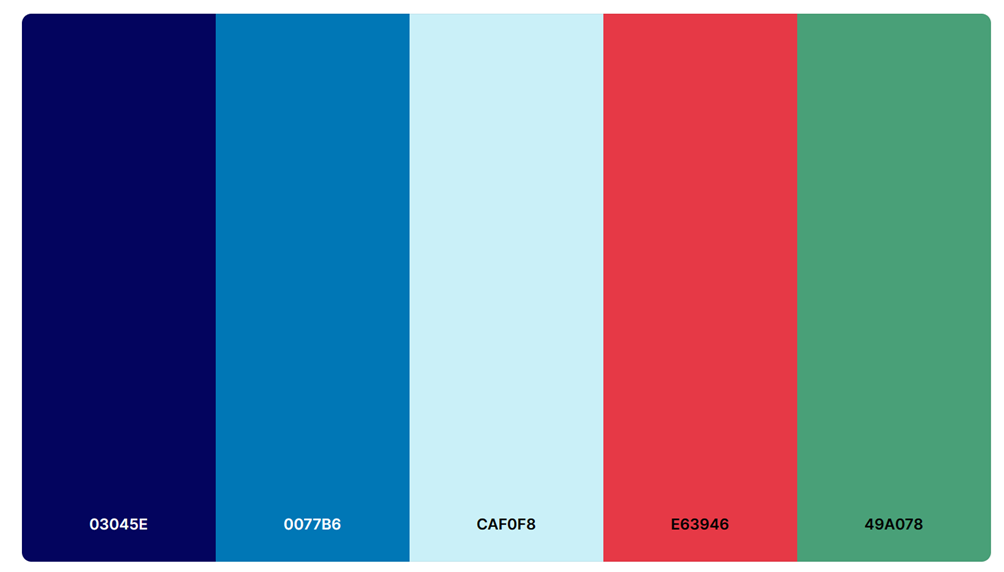
Para concluir, tenemos la parte inferior que tiene un botón de volver al menú principal (3) o el botón más importante de este submenú, que es el botón de iniciar el juego (4)

## Código de colores

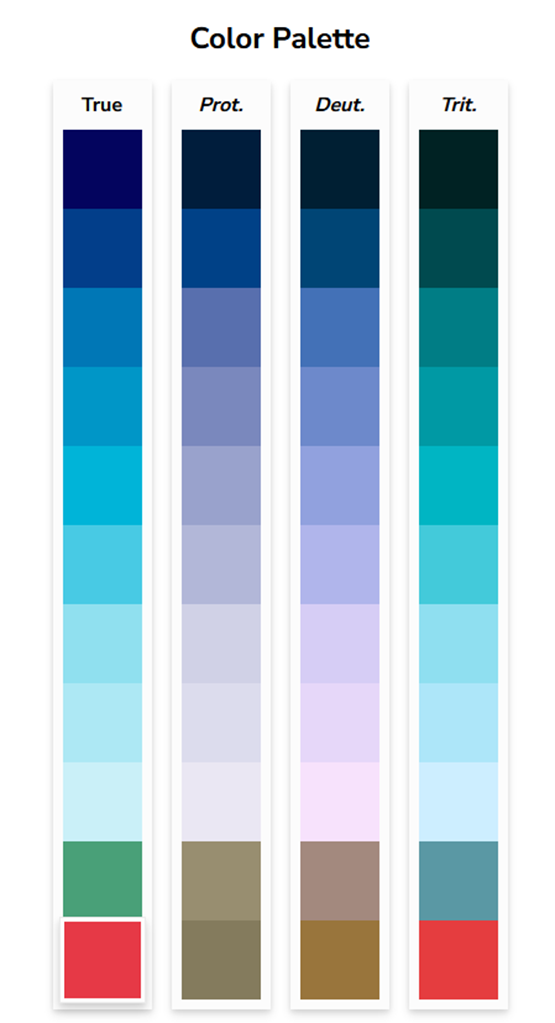
A la hora de diseñar una página web, los colores juegan un papel vital en la experiencia de usuario. Partiendo que unos colores adecuados sirven para hacer una navegación más intuitiva, utilizable y hasta que sirven para que personas con problemas de accesibilidad como el daltonismo puedan usar la aplicación web. Por tanto, a la hora de diseñar este prototipo se tuvo en cuenta este elemento de cara a la accesibilidad.

Para lograr este objetivo se ha asegurado de que todos los elementos existentes en la interfaz sigan una paleta de colores. Por suerte, existen páginas que o bien poseen un catálogo con diferentes paletas de colores o bien tienen generadores que crean una paleta en segundos, como es el caso de Coolors[T]. Aparte de tener ambas opciones previamente mencionadas, posee un analizador de colores y un inspector de contraste que son herramientas que ayudan a la hora de elegir colores y comprobar como interactúan unos colores con otros o cómo los distintos tipos de daltonismo puede percibir el color o inclusive la paleta.

 Para mantener la cantidad de colores bastante limitada se ha usado una paleta de colores mayoritariamente monocromática[U], degradando la intensidad de los mismos.

Adicionalmente se han usado un par de colores para generar contraste entre esta base, permitiendo identificar elementos importantes de forma intuitiva, como pueden ser enemigos. [V]

Finalmente usando otra herramienta web como es Coloring for Colorblindness[W] podemos comprobar cómo se ven realmente la paleta completa con los distintos tipos de daltonismo.



# Bibliografía

Benkler, Y. (2012). *El Pingüino y el Leviatán: Por qué la cooperación es nuestra arma más valiosa para mejorar el bienestar de la sociedad.*

Velasco, J. G. (2009). *Energías renovables.* Editorial Reverte