Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Creación de videojuegos para la gamificación de una asignatura

Autor:
D. Javier Gutiérrez Rodríguez

Tutores:
Dr. Carlos Enrique Vivaracho Pascual
D. Alma Pisabarro Marrón

Resumen

Este Trabajo de Fin de Grado consiste en el desarrollo de dos videojuegos para la gamificación de la asignatura Fundamentos de Programación. Forma parte de un proyecto de investigación que recogerá datos para comprobar el impacto en los estudiantes. Estos videojuegos estan asociados a conceptos sobre los tipos de datos y la recursividad, ayudando al aprendizaje de ambos.

Tabla de Contenidos

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

Este proyecto consiste en el desarrollo de dos videojuegos mediante el uso de Unity. Especificamente, se van a plantear e implementar dos juegos serios que servirán como apoyo al desarrollo de la asignatura de primero Fundamentos de Programación de Grado en Ingeniería Informática. Este proyecto forma parte de una plataforma con diversos juegos con el objetivo final de poder gamificar el temario completo de la asignatura. Los distintos juegos se integran y lanzan desde una plataforma común, desarrollada en un TFG previo.

La finalidad de este proyecto es seguir explorando la utilidad de los juegos como plataformas de aprendizaje. La gamificación consiste en el uso de mecánicas de juego en un contexto no lúdico con el fin de conseguir determinados objetivos. En este caso, se trata del ámbito educativo y lo utilizarémos para mejorar el aprendizaje. Más adelante entraremos en detalles sobre sus ventajas e incovenientes, pero es una corriente que esta adquiriendo gran popularidad en el ámbito de la enseñanza los últimos años debido a su buen funcionamiento.

En nuestro caso, la gamificación de la asignatura de apoyará con dos minijuegos desarrollados en Unity. El primer juego, **Caída de Datos**, se centra en explicar y representar de manera gráfica los distintos tipos de datos, así como sus distintos rangos y explorando temas como el tipo de retorno de una función. En el segundo juego, **Fiesta Recursiva**, se intentan explicar conceptos de la recursión, entendiendo las precondiciones y los casos básicos de una función recursiva.

Como se han desarrollado dos minijuegos distintos, en este documento se tratarán por separado los aspectos correspondientes al planteamiento e implementación de cada uno de ellos.

1.1. Motivación

El juego ha desempeñado un papel importante a lo largo de la historia de la humanidad, además de como elemento lúdico tambien ha servido de elemento creador y una manera de expresarse uno mismo.

Su utilidad como plataforma de aprendizaje está bien acreditada y hay muchos estudios de diversos autores que han explorado este tema. Pero estamos acostumbrados a retringir su uso a un breve periodo de tiempo en la infacia. Por eso, en algún momento el aprendizaje pasa de ser algo apasionante, a convertirse en algo monótono. Es aqui donde entran los juegos serios, que son capaces de aportarnos diversión y motivación a la vez que sirven como vehículo del aprendizaje.

En este proyecto se han itentado plantear unas mecánicas que sean capaces de fomentar cierta dinámicas en el juego como son el reto, la curiosidad y la exploración. De esta forma, el aprendizaje mediante el juego puede ser una experiencia enriquecedora para el alumno a la vez que se estimula su creatividad.

La plataforma del proyecto en conjunto con los juegos serios servirá tanto a los estudiantes como docentes para descubrir nuevas posibilidades en la enseñanza. Ambos se verán beneficiados de esta experiencia, los alumnos podrán mejorar sus resultados y asimilar mejor conceptos de la asignatura de una manera más divertida y práctica. Los profesores podrán controlar el desempeño de los alumnos y estar al tanto de sus progresos, a la vez que siguen explorando opciones para intentar mejorar el proceso de aperendizaje los alumnos.

1.2. Objetivos

El objetivo principal de este proyecto es la creación y puesta en marcha de dos minijuegos en Unity para ayudar al aprendizaje de conceptos relacionados con los tipos de datos simples y la recursividad dentro del ámbito de la programación. Como objetivos más concretos se plantean:

- Poner en marcha e integrar ambos juegos en la plataforma del proyecto de enseñanza.
- Cada juego dispondrá de varios niveles que supondrán una progresión en los conceptos de la asginatura, aumentando la dificultad progresivamente.
- Se obtendrá una puntuación que servirá para evaluar el rendimiento del jugador, pero siendo secundario al aprendizaje.
- Los juegos deben ser capaces presentar un reto que motive a los alumnos a seguir jugando por diversión.

1.3. Contenido de la Memoria

El contenido de la memoria estará formado por 7 capítulos, en cada unos se tratarán los temas que se desglosan a continuacion:

- En el capítulo 2 se va explorar en profundidad el tema de la gamificación en el ámbito del aprendizaje así como aportar claridad en conceptos básicos intrinsecos al desarrollo de videojuegos.
- En el capítulo 3 se describen las herramientas y tecnologías empleadas en este proyecto, haciendo especial incapie en Unity.
- En el capítulo 4 se expone la metodología que se ha adoptado para llevar a cabo este proyecto así como el planteamiento inicial acompañado del seguimiento, gestión de riesgos y presupuesto.
- En los capítulos 5 y 6 se describe el desarrollo de los videojuegos, de formas separada en sus respectivos capítulos. Cada uno contará con GDD, análisis, diseño y pruebas.
- En el capítulo 7 se realizan unas breves conlusiones generales por el proyecto, a la vez que se exploran posibles expansiones futuras de este trabajo.

Capítulo 2

Contexto

Los juegos llevan entren nosotros desde hace miles de años, los videojuegos, más en concreto, desde hace varias decadas. Siempre han sido una experiencia disfrutada por personas de todas las edades y tienen un papel importante en nuestra sociedad.

A lo largo de los últimos años la industría de los videojuegos no ha parado de crecer hasta el punto de ser una de las más importantes a día de hoy. Existen muchísimos productos y multitud de empresas en el sector que buscan innovar constantemente dando lugar a nuevos tipos de juegos. En la actualidad, millones de personas hacen uso de los videojuegos en el día a día, siendo no solo un entretenimiento sino tambien un punto de encuentro.

Aprovechando este aumento del impacto de los videojuegos, han surgido tambien nuevas usos para ellos, en este caso, como una herramienta educativa que sirve para adquirir conocimientos y desarrollar hablidades en un entorno lúdico. De esta manera, se aborda uno de los principales problemas que se encuentra en las aulas, motivar a los alumnos. Incluir juegos en el aprendizaje, nos permite hacer que sea algo divertido e interesante para el alumno gracias a las características intrinsecas al juego.

Esto se conoce como la ludificación o gamificación [6], que consiste en el uso de técnicas, elementos y dinámicas propias de los juegos y el ocio en actividades no necesariamente recreativas, con el fin de potenciar la motivación, así como de reforzar la conducta para solucionar un problema, mejorar la productividad, obtener un objetivo, activar el aprendizaje y evaluar a individuos concretos.

La gamificación puede ser usada en muchos ámbitos, pero en este caso nos centraremos en el ámbito educativo, concretamente, mediante el uso de juegos serios que son aquellos cuyo objetivo principal es el aprendizaje, en lugar de ser el entretenimiento solamente.

El modelo de juego funciona porque consigue motivar a los alumnos, desarrollando un mayor compromiso de las personas, e incentivando el ánimo de superación. Para ello se utilizan una serie de técnicas, mecánincas y dinámicas extrapoladas de los juegos. Debido a su caracter lúdico, facilita la interiorización de conceptos de una forma más divertida, generando una experiencia positiva en el alumno [3].

La gamificación tiene muchas ventajas en el campo docente [4], entre ellas:

- **Aumentar la motivación de los alumnos:** la gamificación nos permite crear experiencias divertidas y retos que consiguen aumentar la motivación de los alumnos haciendo uso de la predisposición psicológica del ser humano a jugar. Tambien los elementos competitivos y de superación ayudan a esto.
- **Fracasar no es malo:** al estar en un ambiente lúdico los alumnos pueden aprender mediante la repetición sin llegar a ver el fracaso como algo malo y simplemente un paso más en el aprendizaje.
- **Favorecen la socialización:** el juego como hemos visto antes se caracteriza por ser social, dentro de la gamificacion tambien existen elementos que hacen que el alumno tenga que cooperar y competir con otros jugadores. De esta manera se favorece el trabajo en equipo y la creación de un espacio de experiencias compartidas.
- **Dsarrolla la creatividad:** en los juegos puedes enfrentarte a problemas conocidos de maneras poco convencionales, esto hace que jugar requiera soluciones imaginativas estimulando así la creatividad.
- **Realimentación en tiempo real:** en la gamificación se pueden obetener datos de la progresión en el juego, obteniendo así medidas claras sobre los conocimientos adquiridos por los alumnos.
- **Se puede ver la utilidad real de los conocimientos:** el alumno puede ver de manera directa las consecuencias negativas y positivas de sus decisiones en el juego, ayudando a entender de manera directa los conceptos involucrados en el juego. La posibilidad de aplicarlo de manera práctica es una forma muy eficaz de asentar conocimientos.

Tambien exiten algunos incovenientes provocados por el uso de juego en el aula y la gamificación, algunos de ellos [3]:

- **Puede centrar al jugador en ganar:** el caracter competitivo de los juegos y los posibles objetivo a lograr pueden hacer que el jugador pierda de vista el aprendizaje y centre sus esfuerces en conseguir la mayor puntuación posible. Para poder combatir este problema es necesario diseñar juegos con mecánicas que favorezcan siempre en aprendizaje a pesar de que la meta final del jugador sea ganar.
- **Motivación pasajera:** los jugadores pueden obtener una motivación inicial muy elevada pero que puede verse rapidamente mermada si no consiguen alcanzar buenos resultados o no consiguen completar el juego. Tambien puede darse el caso de que la dificultad del juego sea insuficiente y por lo tanto los jugadores no tengan suficiente motivación para aprender los conceptos tratados en el juego. Como medidas para paliar este problema se debe intentar que el diseño de niveles favorezca que el sean asequibles de completar pero requiera cierta practica alcanzar una buena puntuación.

2.1. Concpetos Relacionados con el Desarrollo de Videojuegos

En esta sección se van a introducir ciertos conceptos relacionados con el desarrollo de los videojuegos y que han sido una parte importante de este proyecto.

2.1.1. Documento de Diseño del Juego

El Documento de Diseño del Juego o GDD (Game Design Document) es una pieza clave en el desarrollo de un videojuego e incluye todos los aspectos básicos relaccionados con su diseño. Su objetivo principal es comunicar de manera clara los detalles tecnicos y creativos del juego y servir como piedra de toque a lo largo del desarrollo.

Este documento no debe ser muy extenso y debe priorizar la claridad, al igual que el juego estará en constante evolución incluyendo los cambios que sean necesarios. A pesar de su importancia no sustituye a reuniones de equipo o documentos con más detalle en ciertos apartados. Este documento tiene distinos apartados, que no son fijos, y que dependerán tanto del juego que se este desarrollando como del esquema que se use.

Para este proyecto se ha decidido que los documentos sigan el esquema de Tracy Fullerton [9], que consta de los siguientes apartados:

- **Visión del juego:** en este apartado se intenta capturar la esencia del juego y transmitirla al lector de una manera clara y precisa. Es importante comentar los detalles que hacen único al juego y las mecánicas principales.
- **Audiencia, plataforma y marketing:** aqui se describe el publico objetivo del juego, las plataformas en que podrá ser jugado así como un estudio de mercado para el juego comparandolo con titulos parecidos y las expectativas de ventas. Dependiendo del juego esta sección puede variar y omitir parte como el estudio de mercado.
- **Mecánicas y dinámicas:** en este apartado se desglosan las mecánicas principales del juego y las dinámicas que ofrecen a la hora de jugarlo. Tambien se detallan otros temas como los controles, condición de victoria, niveles y modos de juego presentes.
- **Personajes:** aqui se detallan los distintos personajes que aparezcan en el juego y los diversos tipos de personaje que puedan aparecer diferenciando entre jugables y no jugables.
- **Historia y mundo:** en esta sección se desarrollan los distintos aspectos de la trama si es que existe y pueden definirse detalles clave del mundo en el que se desarrolla.
- **Apartado técnico y multimedia:** por último se necesitar listar todos los elementos multimedia que deben ser creados para el juego incluyendo personajes, animaciones, música, entornos etc. En cuanto al aspecto tecnico se definirá que tecnología se usará y si es necesario desarrollar una nueva así como las herramientas empleadas.

2.1.2. Pruebas Beta

En cualquier desarrollo las pruebas van a ser una parte muy importante y necesaria puesto que necesitamos saber si nuestro sistema responde correctamente a todas las acciones del usuario y detectar posibles fallos.

En el desarrollo de los videojuegos es una tarea fundamental puesto que la complejidad de tantas mecánicas funcionando en conjunto puede dar lugar a numerosos errores dificiles de detectar. Para los desarrolladores es casi imposible de detectar muchos de ellos y es donde surge la necesidad de realizar pruebas beta.

Este tipo de pruebas consiste en dejar probar el producto a los usuarios finales. La gente que realiza estas pruebas son llamados betatester. Normalmente se suelen realizar con distintas versiones del juego en pequeños entornos y en un punto más avanzado y depurado con un grupo mayor de usuarios.

Capítulo 3

Herramientas Utilizadas

3.1. Unity

En el mundo actual los videojuegos han aumentado su presencia en nuestras vidas para convertirse en una parte importante en la vida de muchas personas. Todo esto tambien se ha visto reflejado en un mayor interés en su creación. Por ello, han cobrado una gran importancia los motores de desarrollo de videojuegos que consisten en un entorno de herramientas que nos permiten desarrollar los principales componentes de un juego tales como renderizado, animación, modelado, sonido y otros aspectos.

Uno de los requisitos de este proyecto era usar Unity, una de las plataformas de desarrollo más populares hoy en día y que acoge uno de los ecosistemas de desarrolladores mas amplios en la industria. Ofrecen una amplia gama de alternativas para todo tipo de usuarios, desde un uso educativo hasta profesional. En nuestro caso, la plataforma principal del proyecto se desarrolló en Unity, por lo tanto, es necesario usarlo para integrar de manera adecuada los videojuegos.

La idea de Unity nace en 2004 de la mano de Unity Technologies con un objetivo claro: democratizar el desarrollo de videojuegos. Comienzan la creación de un motor de videojuegos que pequeñas y grandes empresas puedan utilizar por igual. Un entorno amigable donde puedan trabajar programadores, artistas y diseñadores sin tener que crear herramientas especificas para cada apartado. Desde entonces no ha parado de crecer hasta convertirse en uno de los motores multiplataforma más potentes y utilizados del mercado.

Una de las características más importantes de Unity [8] es que se trata de un motor multiplataforma. Esto quiere decir que con una sola herramienta podemos desarrollar videojuegos para las principales plataformas del mercado realizando solo ciertos ajustes y no tener que crearlo desde cero con otra herramienta. Entre estas plataformas figuran ordenadores, consolas, móviles y plataformas de realidad virtual, es decir con una sola plataforma tenemos la posibilidad de desarrollar para varias plataformas sin un aumento de costes y con la posibilidad de alcanzar un mayor público.

Actualmente Unity es usado por miles de compañías para producir sus juegos desde estudios independientes hasta grandes compañías. Algunos de los juegos con más repercusión creados con esta plataforma son: Hollow Knight, Fall Guys: Ultimate Knockout, League Of Legends: Wild Rift o Outer Wilds. Desde Unity Technologies tambien han intentado dar el salto a otros sectores fuera de los videojuegos como son el cine y la animación, uno de los más importantes se trata de una colaboración con Disney Television Animation para crear tres cortos dentro de un proyecto llamado Baymax Dreams.

Además de todo lo expuesto, tambien podemos destacar estas otras características [7]:

- **Documentación:** el entorno de desarrollo Unity dispone de una excelente documentación, una de las mejores que existe en motores de desarrollo de videojuegos. En el manual de Unity, se pueden consultar todos y cada uno de sus distintos apartados, ya sea sobre como actualizar a una versión concreta o incluso guías profesionales para realizar tareas de un nivel muy avanzado. Tambien es posible encontrar un historial de las versiones anteriores de la documentación. De manera que podemos consultar cualquier versión sin necesidad de estar cerrados a la última versión de Unity.

- **Comunidad:** como hemos explicado antes Unity es una de las platformas más populares y gracias a esto posee una gran comunidad por lo que la cantidad de recursos a nuestra disposición es muy abundante. Cualquier tipo de duda que nos pueda surgir encontraremos respuesta con mucha facilidad, acceso a tutoriales explicando diversas tecnicas, la posibilidad de ver proyectos de otras personas. Podemos encontrar todo esto directamente en los foros de Unity o buscando diversas fuentes por internet.

- **Unity Asset Store:** desde Unity han creado una de las mayores tiendas de recursos para videojuegos que podemos encontrar en internet. En ella podemos encontrar todo tipo de recuros desde modelos, texturas, efectos, ampliaciones del editor, servicios y mucho más. Aparte exite una gran cantidad de contenido gratuito que puedo ayudar a pequeños proyectos. Tambien sirve para que creadores de recursos pueden tener un mercado a su disposición en el que vender su trabaja a otras desarrolladores.

- **Herramientas profesionales:** incluso con las versiones gratuitas de Unity tenemos acceso a las mismas herramientas que un desarrollador profesional, esto nos permite crear videojuegos con una gran calidad gráfica y contenido sin necesiad de hacer un gran desembolso.

- **Facilidad de uso:** una de las características más llamativas de Unity es su facilidad de uso incluso si no estas acostumbrado a este tipo de motores de desarrollo. Posee una interfaz muy simple e intuitiva que nos permite visualizar los cambios en tiempo real, muchas acciones pueden realizarse desde interfaces sin necesidad de saber programar. Puedes centrarte en las partes que más te interesen sin requerir un conocimiento completo de todo el sistema.

3.1.1. Scripting con UnityEngine

A la hora de desarrollar juego en Unity, una de las maneras fundamentales de trabajar es mediante el uso de scripts. Los componentes integrados de Unity son muy versátiles, pero para poder llegar más allá e implementar las características propias de tu juego se necesita realizar scripts. Unity soporta dos lenguajes nativamente: C#, un lenguaje estándar de la industria similar a Java o C++ y UnityScript, un lenguaje diseñado especificamente para uso con Unity y modelado tras JavaScript.

Para el desarrollo de estos scripts Unity ofrece una librería llamada UnityEngine, que será fundamental para poder manejar los elementos del juego. En la figura 3.1 se muestra la estructura básica de un juego en Unity.

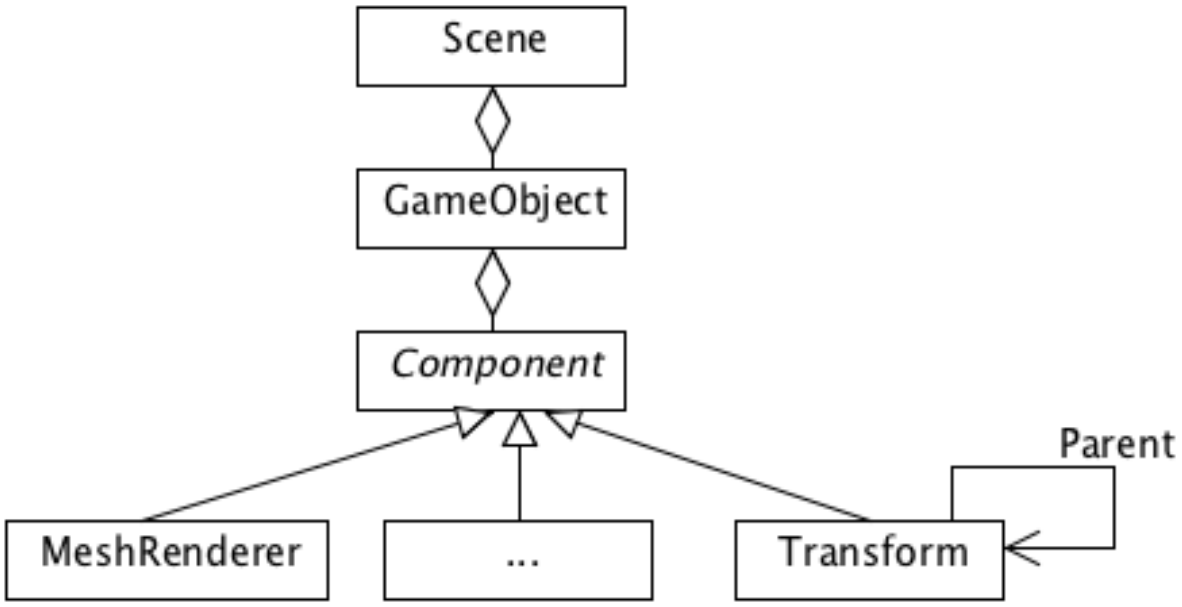

Figura 3.1: Diagrama estructura de juego en Unity

El elemento principal que compone un juego en Unity son las escenas. Las escenas sirven de contenedor donde trabajar con todo el contenido del juego, pueden contener todo el juego o una parte. Por ejemplo, si se esta construyendo un juego simple se puede usar una sola escena, pero en un juego más complejo puede ser necesario usar varias escenas, cada una con un decorado, personajes, obstáculos, e interfaz. De esta manera, se puede estructurar el juego de una manera sencilla y separar las distintas partes. A partir de este punto todos los elementos que se añadan a la escena seran GameObject, que es el elemento más importante de UnityEngine.

Cada objeto de un juego es un GameObject, desde personajes, hasta luces, cámaras y efectos especiales. Por si mismos no son capaces de realizar ninguna acción, pero actúan como contenedores de los componentes que son los encargados de implementar las distintas funcionalidades. Estos componentes son la clase base que nos permite modificar el comportamiento de los GameObject. Para poder conseguir la funcionalidad deseada se debe añadir distintas combinaciones de componentes. Unity incluye una gran cantidad de componentes por defecto pero es posible crear componentes propios si fuese necesario.

Uno de los componentes más importantes es el componente Transform que siempre esta presente en los GameObject y no puede ser eliminado. Este componente se encarga de definir la posicion, rotación y escala del GameObject dentro de la escena. Existen muchos tipos de componentes en Unity, algunos de los más importantes son los Renders que se encargan de renderizar las distintas imagenes y texturas en el mundo tanto 2D como 3D. Tambien existen componentes que nos permiten aplicar fisicas y colisiones a nuestros GameObject, algunos de los más usados son RigidBody y Collider.

Una vez creados todos estos elementos vamos a necesitar definir su comportamiento, aquí es donde entran en juego los scripts, que son la parte programable de Unity. Cualquier juego por muy simple que sea hará uso de scripts para para poder crear las mecánicas del juego. Desde los controles, pasando por efectos gráficos, físicas y cualquier cosa que necesite un juego deberá ser configurada con ayuda de scripts. Los scripts tambien son componentes que implementan la clase MonoBehaviour y eso les permite ser añadidos a un GameObject.

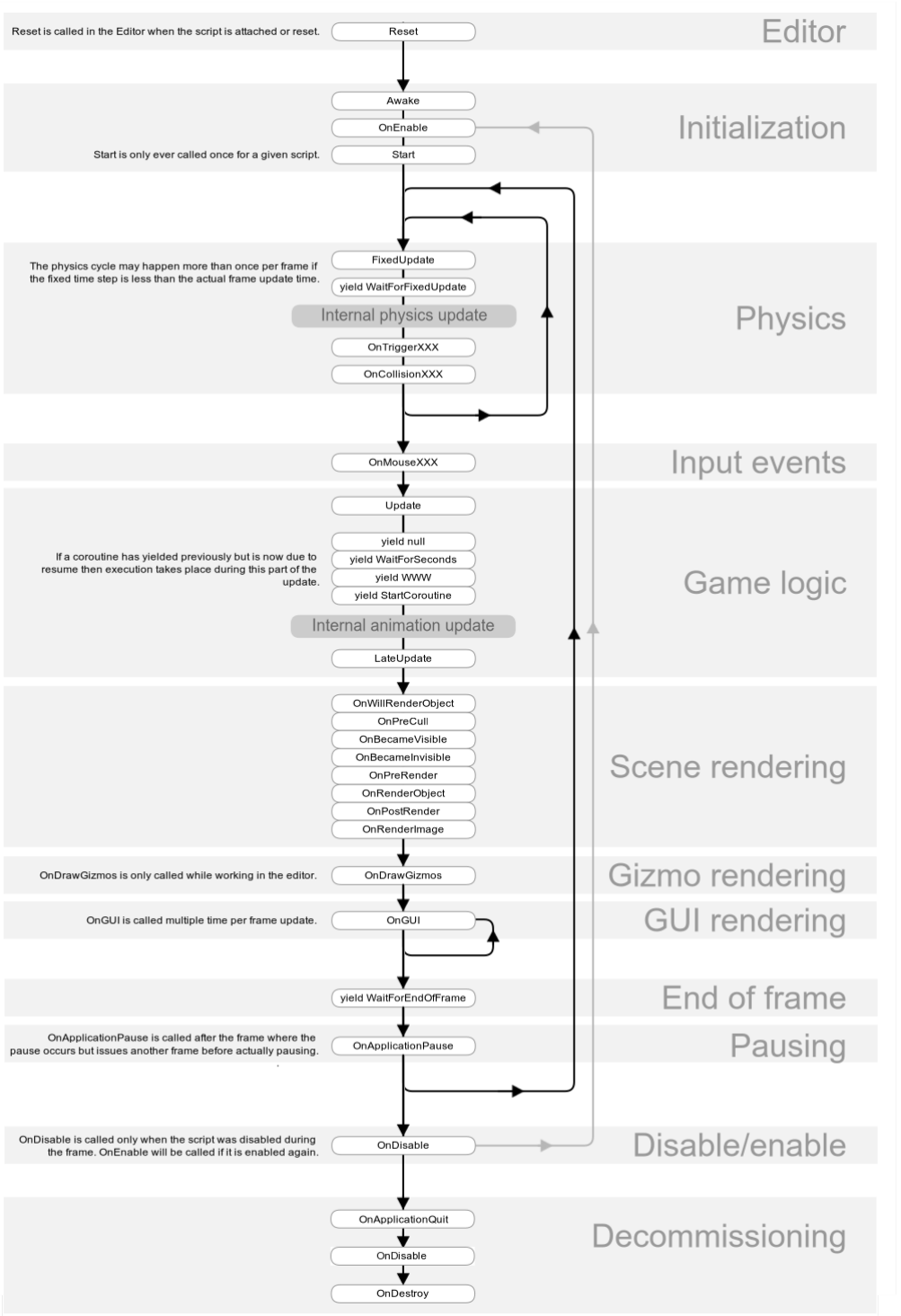

Figura 3.2: Flujo de ejecución de funciones en Unity

Estos script pueden ser ejecutados por eventos realizados por el jugador pero tambien existen funciones predeterminadas de Unity que los ejecutan. En la figura 3.2 se puede observar el flujo de ejecucion, las llamadas que se realizan con sus condiciones de entrada y salida.

Algunas de las funciones más importantes son:

- **Start:** se llama en la primera actualización de frame si el script esta activo.
- **Update:** se llama una vez por frame y es la función principal.
- **OnDestroy:** se llama despues de todas las actulizaciones de frame para el último frame de existencia del objeto.

Capítulo 4

Plan de Proyecto

4.1. Metodología

4.1.1. Descripción

Para el desarrollo de este proyecto se ha decidido utilizar un desarrollo en cascada [1] (figura 4.1) como marco de trabajo. El modelo en cascada es un proceso de desarrollo secuencial, en el que el software se concibe como un conjunto de etapas que se ejecutan una tras otra. Al contrario que en los modelos iterativos, cada una de estas etapas solo se ejecuta una vez. Los resultados de cada etapa sirven como punto de partida para la siguiente.

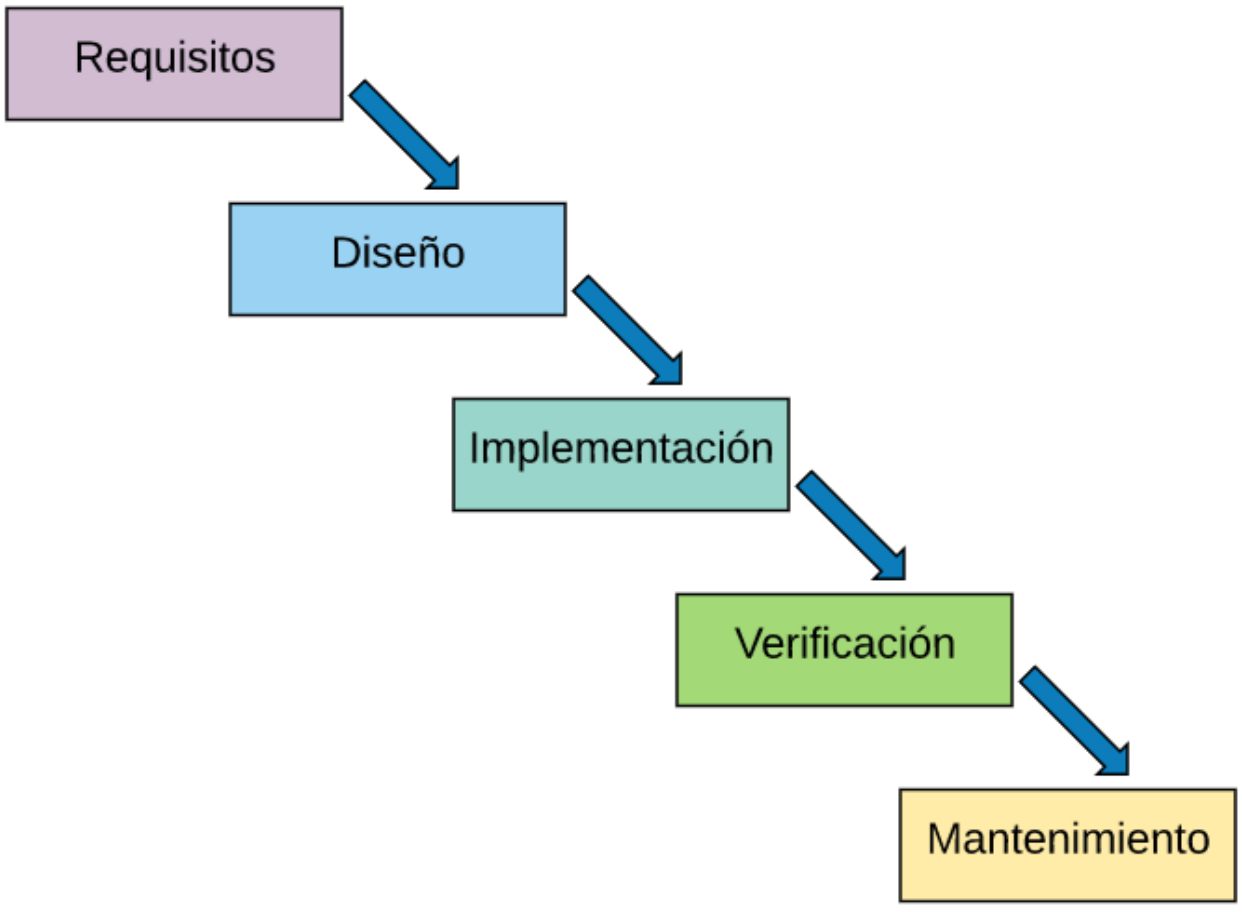

Figura 4.1: Desarrollo en cascada

4.1.2. Fases

Existen muchas variantes del modelo pero los más habituales son aquellos que dividen el proceso de desarrollo en cinco fases.

Para este proyecto se han incluido ampliaciones de la metodología en cascada añadidiendo funciones iterativas al modelo básico, de esta manera se puede realizar saltos de etapa hacia atrás. Esto nos permite comprobar si los resultados obtenidos coinciden con lo planteado en la etapa anterior y en caso negativo repetir la etapa para alcanzar el resultado deseado.

Se trata de un modelo lineal, por lo que las diferentes fases se ejecutan una tras como en una cascada. Al finalizar cada una de las fases se obtiene un resultado provisional o hito como, por ejemplo, un acuerdo en el alcance del proyecto o una aplicación prototipo.

Análisis

En esta fase se hace un análisis de las necesidades de los usuarios finales para determinar que objetivos debe cubrir sin entrar en detalles técnicos. Por lo tanto, esta es la etapa en la que se recogen la mayoría de los requisitos del software y se identifica el alcance del proyecto, los posibles riesgos que pueden surgir y estimaciones de costes, esfuerzo y tiempo.

Diseño

En esta fase se describe la estructura interna del software y las relacciones entre la entidades que lo componen. Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, de esta manera se obtiene una descripción sobre lo que debe hacer cada una de sus partes y de que manera funcionan en conjunto.

Implementación

En esta fase se procede a programar los requisitos especificados haciendo uso de la estructura definida en la fase anterior. Se hacen uso de prototipos así como de pruebas y ensayos para corregir errores.

Verificación

En esta fase se verifica que todos los componentes del sistema funcionan de manera correcta y que cumplen con los requisitos planteados. El objetivo de esta fase es conocer la calidad del software mediante la realización de pruebas. Esto nos permite encontrar defectos, refinar el código y aumentar la calidad.

Mantenimiento

En esta fase se instala la aplicación en el sistema y se comprueba que funciona bien dentro del entorno en el que se va a usar.

4.1.3. Ventajas

- Se obtiene un proyecto estructurado con unas fases de desarrollo claras y bien definidas

- La documentación del proceso de desarrollo es buena a traves de objetivos definidos.
- Los costes y la carga de trabajo se puede estimar al comenzar el proyecto.
- Los proyecto se pueden representar cronológicamente de forma sencilla.

4.1.4. Inconvenientes

- El usuario final no participa en el proceso de desarrollo hasta que no finaliza la implementación.
- Poco margen para realizar ajustes si ocurre un cambion en los requisitos.
- Los posibles fallos solo se detectan una vez finalizado el proceso de desarrollo.
- Los proyectos complejos no se pueden dividir correctamente en fases claras y bien definifas.

4.2. Plan de gestión de riesgos

En esta sección se van a especificar los riesgos que deben tenerse en cuenta para llevar a cabo el desarrollo de proyecto. Es necesario identificarlos ya que pueden ser un incoveniente en el correcto desarrollo del proyecto. Para ello, en el proceso de análisis de los riesgos se debe obtener el grado de exposición al riesgo. Esto se puede obtener teniendo en cuenta su probabilidad de ocurrir y el impacto que podrían causar. En las figuras 4.1 a 4.8 se muestran los riesgos detectados, su probabilidad e impacto, así como los planes de contingencia previstos para cada uno.

Riesgo 1	**Enfermedad o situaciones excepcionales**
Probabilidad:	Media
Impacto:	Medio
Protección ante el riesgo: Mantener un estado de salud correcto.	
Plan de contingencia: Seguir las recomendaciones adecuadas a la situación.	
Una enfermedad nos impide realizar el trabajo de manera habitual	

Tabla 4.1: Riesgo 1

Riesgo 2	Pérdida de datos o documentación
Probabilidad:	Baja
Impacto:	Alto
Protección ante el riesgo: Mantener una copia de seguridad actualizada del proyecto y usar un sistema de control de versiones.	
Plan de contingencia: Evaluar el impacto de las pérdidas y planificar en consecuencia.	
Una perdida de datos supondría un retraso en el trabajo	

Tabla 4.2: Riesgo 2

Riesgo 3	Pérdida de conexión a internet
Probabilidad:	Media
Impacto:	Medio
Protección ante el riesgo: Disponer de un dispositivo de conexión a internet alternativo y guardar localmente el material de consulta.	
Plan de contingencia: Trabajar en aquellas tareas que no requieran conexión a internet.	
Se produce un corte no planificado en el suministro de internet que no permite realizar las tareas correctamente	

Tabla 4.3: Riesgo 3

Riesgo 4	Cambio en los requisitos
Probabilidad:	Media
Impacto:	Alto
Protección ante el riesgo: Mantener un desarrollo estructurado para facilitar la realización de cambios en el sistema.	
Plan de contingencia: Realizar las modificaciones necesarias en el proyecto y planificar en consecuencia.	
Se produce un cambio en los requisitos del proyecto, lo cuál requiere más tiempo y/o recursos	

Tabla 4.4: Riesgo 4

Riesgo 5	Fallo en la máquina de trabajo
Probabilidad:	Baja
Impacto:	Medio
Protección ante el riesgo: Mantener la máquina en buenas condiciones y controlar todas las anomalías detectadas.	
Plan de contingencia: Conseguir una máquina de repuesto o reparar la actual y montar el entorno de desarrollo en ella.	
Se produce un fallo en la máquina que ralentiza o imposibilita el desarrollo del proyecto	

Tabla 4.5: Riesgo 5

Riesgo 6	Imposibilidad de comunicación con el cliente
Probabilidad:	Media
Impacto:	Medio
Protección ante el riesgo: Adelantarse a los periodos en que una de las dos partes no pueda reunirse obteniendo información suficiente para porder trabajar.	
Plan de contingencia: Realizar aquellas tareas que no requieran consulta con el cliente.	
Por razones ajenas al proyecto es imposible reunirse con el cliente ralentizando el avance del proyecto.	

Tabla 4.6: Riesgo 6

Riesgo 7	Conocimiento insuficiente sobre las tecnologías
Probabilidad:	Media
Impacto:	Alto
Protección ante el riesgo: Investigar el nivel de conocimiento necesario para las tareas con antelación y tener en cuenta el tiempo de aprendizaje en la planificación.	
Plan de contingencia: Replanificar las taresa y dedicar más tiempo al aprendizaje para poder avanzar en el proyecto.	
No se ha dedicado suficiente tiempo para aprender las herramientas y las tecnologías a usar impidiendo desarrollar lo que se requiere en la aplicación.	

Tabla 4.7: Riesgo 7

Riesgo 8	Fallo en el cumplimiento de la planificación temporal
Probabilidad:	Media
Impacto:	Alto
Protección ante el riesgo: Llevar a cabo el cumplimiento semanal de la planificación para mantener el ritmo del proyecto establecido.	
Plan de contingencia: Realizar tareas para recuperar trabajo no realizado anteriormente.	
Debido a las complicaciones surgidas durante el desarrollo del proyecto no se puede cumplir con la planificación.	

Tabla 4.8: Riesgo 8

4.3. Plan de trabajo

En las tablas 4.9 a 4.17 se pueden ver las estimaciones temporales y división de tareas para cada una de las fases. Para este proyecto se han desarrollado dos aplicaciones distintas para las cuales se han producido todas las fases de desarrollo excepto la etapa inicial que es común a ambas. La cantidad de horas por días es flexible, sin una relación directa entre días transcurridos y horas invertidas en el proyecto.

Tarea	Horas	Inicio	Fin
TOTAL	55	1/3/2020	15/3/2020
Inicio	0	1/3/2020	1/3/2020
Planteamiento inicial del problema	1	1/3/2020	3/3/2020
Definición de requisitos	2	1/3/2020	5/3/2020
Analisis de viabilidad del proyecto	5	1/3/2020	5/3/2020
Estimación de costes	1	1/3/2020	3/3/2020
Calendarización	4	1/3/2020	6/3/2020
Definición de las tareas a desarrollar	10	5/3/2020	10/3/2020
Investigación sobre la gamificación	8	5/3/2020	15/3/2020
Formación en Unity y C	20	5/3/2020	15/3/2020
Preaparar el entorno de desarrollo	5	10/3/2020	15/3/2020
Fin	0	15/3/2020	15/3/2020

Tabla 4.9: Planificación de la fase inicial

Tarea	Horas	Inicio	Fin
TOTAL	46	15/3/2020	31/3/2020
Inicio	0	15/3/2020	15/3/2020
Definición de requisitos del primer juego	2	15/3/2020	16/3/2020
Proceso creativo para definir el juego	20	15/3/2020	25/3/2020
Realización del GDD	4	15/3/2020	25/3/2020
Formación sobre Unity	15	20/3/2020	28/3/2020
Analisis de casos de uso	3	25/3/2020	31/3/2020
Analisis del modelo de dominio	2	25/3/2020	31/3/2020
Fin	0	31/3/2020	31/3/2020

Tabla 4.10: Planificación de la fase de analisis del primer juego

Tarea	Horas	Inicio	Fin
TOTAL	13	31/3/2020	5/4/2020
Inicio	0	31/3/2020	31/3/2020
Investigación de patrones de diseño en Unity	10	31/3/2020	4/4/2020
Arquitectura del juego	3	2/4/2020	5/4/2020
Fin	0	5/4/2020	5/4/2020

Tabla 4.11: Planificación de la fase de diseño del primer juego

Tarea	Horas	Inicio	Fin
TOTAL	48	5/4/2020	20/4/2020
Inicio	0	5/4/2020	5/4/2020
Configuración de Unity	2	5/4/2020	6/4/2020
Obetención y creación de sprites	8	5/4/2020	10/4/2020
Implementación de las funcionalidades	25	5/4/2020	15/4/2020
Implementación de la interfaz de usuario	10	12/4/2020	17/4/2020
Implementación de la conexión con la plataforma web	3	15/4/2020	20/4/2020
Fin	0	20/4/2020	20/4/2020

Tabla 4.12: Planificación de la implementación del primer juego

Tarea	Horas	Inicio	Fin
TOTAL	30	20/4/2020	5/5/2020
Inicio	0	20/4/2020	20/4/2020
Pruebas individuales de funcionalidad	10	20/4/2020	25/4/2020
Pruebas sobre la aplicación completa	10	25/4/2020	30/4/2020
Pruebas en la plataforma web	5	30/4/2020	4/5/2020
Prueba presencial con alumnos	2	4/5/2020	4/5/2020
Elaboracion del videotutorial	3	4/5/2020	5/5/2020
Fin	0	5/5/2020	5/5/2020

Tabla 4.13: Planificación de las pruebas del primer juego

Tarea	Horas	Inicio	Fin
TOTAL	46	5/5/2020	20/5/2020
Inicio	0	5/5/2020	5/5/2020
Definición de requisitos del primer juego	2	5/5/2020	6/5/2020
Proceso creativo para definir el juego	20	5/5/2020	15/5/2020
Realización del GDD	4	5/5/2020	15/5/2020
Formación sobre Unity	15	10/5/2020	18/5/2020
Analisis de casos de uso	3	15/5/2020	20/5/2020
Analisis del modelo de dominio	2	15/5/2020	20/5/2020
Fin	0	20/5/2020	20/5/2020

Tabla 4.14: Planificación de la fase de analisis del segundo juego

Tarea	Horas	Inicio	Fin
TOTAL	13	20/5/2020	25/5/2020
Inicio	0	20/5/2020	20/5/2020
Investigación de patrones de diseño en Unity	10	20/5/2020	24/5/2020
Arquitectura del juego	3	22/5/2020	25/5/2020
Fin	0	25/5/2020	25/5/2020

Tabla 4.15: Planificación de la fase de diseño del segundo juego

Tarea	Horas	Inicio	Fin
TOTAL	48	25/5/2020	10/6/2020
Inicio	0	25/5/2020	25/5/2020
Configuración de Unity	2	25/5/2020	26/5/2020
Obetención y creación de sprites	8	25/5/2020	30/5/2020
Implementación de las funcionalidades	25	25/5/2020	5/6/2020
Implementación de la interfaz de usuario	10	2/6/2020	7/6/2020
Implementación de la conexión con la plataforma web	3	5/6/2020	10/6/2020
Fin	0	10/6/2020	10/6/2020

Tabla 4.16: Planificación de la implementación del segundo juego

Tarea	Horas	Inicio	Fin
TOTAL	30	10/6/2020	25/6/2020
Inicio	0	10/6/2020	20/6/2020
Pruebas individuales de funcionalidad	10	10/6/2020	15/6/2020
Pruebas sobre la aplicación completa	10	15/6/2020	20/6/2020
Pruebas en la plataforma web	5	20/6/2020	24/5/2020
Prueba presencial con alumnos	2	24/6/2020	24/5/2020
Elaboracion del videotutorial	3	24/6/2020	25/5/2020
Fin	0	25/6/2020	25/6/2020

Tabla 4.17: Planificación de las pruebas del segundo juego

4.3.1. Seguimiento

En general, las estimaciones del proyecto han sido algo optimistas y los tiempos finales se han alejado de la fecha estimada, debido principalmente a un fallo en el cumplimiento de la planificación temporal. Las estimaciones de horas dedicadas han sido más certeras, excepto algún incoveniente. Pero el plazo de un año máximo que de había propuesto se ha consegudio cumplir.

Todas las fechas se han visto en general trastocadas por la situación actual de pandemia, lo cual era un riesgo imprevisto y que ha afectado más de lo esperado a este proyecto.

En la fase inicial se produce un retraso considerable en su inicio, pero casi todas las tareas son llevadas a cabo dentro del tiempo estimado excepto un ligero aumento en la formación en Unity que llevo más tiempo del esperado (tabla 4.18).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	55	61	1/3/2020	15/3/2020	1/4/2020	15/4/2020
Inicio	0	0	1/3/2020	1/3/2020	1/4/2020	1/4/2020
Planteamiento inicial del problema	1	2	1/3/2020	3/3/2020	1/4/2020	3/4/2020
Definición de requisitos	2	2	1/3/2020	5/3/2020	1/4/2020	5/4/2020
Analisis de viabilidad del proyecto	5	5	1/3/2020	5/3/2020	1/4/2020	5/4/2020
Estimación de costes	1	1	1/3/2020	3/3/2020	1/4/2020	3/4/2020
Calendarizacion	4	4	1/3/2020	6/3/2020	1/4/2020	6/4/2020
Definición de las tareas a desarrollar	10	10	5/3/2020	10/3/2020	5/4/2020	10/4/2020
Investigación sobre la gamificación	8	8	5/3/2020	15/3/2020	5/4/2020	15/4/2020
Formación en Unity y C	20	25	5/3/2020	15/3/2020	5/4/2020	15/4/2020
Preaparar el entorno de desarrollo	5	5	10/3/2020	15/3/2020	10/4/2020	15/4/2020
Fin	0	0	15/3/2020	15/3/2020	15/4/2020	15/4/2020

Tabla 4.18: Seguimiento de la fase inicial

En esta primera fase de analisis centrada en el primer juego es donde se producen los primeros retrasos (tabla 4.19) debido a que el proceso creativo para definir el juego es más complicado de lo esperado, incluso es necesarios dejar esta idea de gamificación para más adelante y centrarse en otra más sencilla. Una vez hecho el cambio se llega rápido a un acuerdo tanto de requisitos como de concepto de juego que encaja con el proyecto. El resto es realizado con bastante rapidez.

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	46	67	15/3/2020	31/3/2020	15/4/2020	31/4/2020
Inicio	0	0	15/3/2020	15/3/2020	15/4/2020	15/4/2020
Definición de requisitos del primer juego	2	2	15/3/2020	16/3/2020	15/4/2020	16/4/2020
Proceso creativo para definir el juego	20	35	15/3/2020	25/3/2020	15/4/2020	30/6/2020
Realización del GDD	4	5	15/3/2020	25/3/2020	15/4/2020	30/6/2020
Formación sobre Unity	15	20	20/3/2020	28/3/2020	20/4/2020	28/4/2020
Analisis del modelo de dominio	3	3	25/3/2020	31/3/2020	25/4/2020	30/6/2020
Analisis de casos de uso	2	2	25/3/2020	31/3/2020	25/4/2020	30/6/2020
Fin	0	0	31/3/2020	31/3/2020	31/4/2020	30/6/2020

4

Tabla 4.19: Seguimiento de la fase de analisis del primer juego

La fase de diseño del primer juego no tenía demasiada carga y se ha ajustado a los tiempos previstos (tabla 4.20).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	13	13	31/3/2020	5/4/2020	30/6/2020	5/7/2020
Inicio	0	0	31/3/2020	31/3/2020	30/6/2020	30/6/2020
Investigación de patrones de diseño en Unity	10	10	31/3/2020	4/4/2020	30/6/2020	4/7/2020
Arquitectura del juego	3	3	2/4/2020	5/4/2020	2/7/2020	5/7/2020
Fin	0	0	5/4/2020	5/4/2020	5/7/2020	5/7/2020

Tabla 4.20: Seguimiento de la fase de diseño del primer juego

La fase de implementación apenas se desvía de los tiempos previstos y solo algunas tareas como la conexión con la plataforma web llevan algo más de tiempo del esperado por problemas ajenos al proyecto (tabla 4.21).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	48	52	5/4/2020	20/4/2020	5/7/2020	20/7/2020
Inicio	0	0	5/4/2020	5/4/2020	5/7/2020	5/7/2020
Configuración de Unity	2	2	5/4/2020	6/4/2020	5/7/2020	6/7/2020
Obetención y creación de sprites	8	12	5/4/2020	10/4/2020	5/7/2020	10/7/2020
Implementación de las funcionalidades	25	25	5/4/2020	15/4/2020	5/7/2020	15/7/2020
Implementación de la interfaz de usuario	10	8	12/4/2020	17/4/2020	12/7/2020	17/7/2020
Implementación de la conexión con la plataforma web	3	5	15/4/2020	20/4/2020	15/7/2020	20/7/2020
Fin	0	0	20/4/2020	20/4/2020	20/7/2020	20/7/2020

Tabla 4.21: Seguimiento de la fase de implementación del primer juego

Las pruebas del primer juego tambien han ido de acuerdo con las estimaciones iniciales y al no ser un proyecto de grán tamaño no ha habido contratiempos (tabla 4.22). La prueba presencial se tuvo que retrasar por la pandemia.

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	30	30	20/4/2020	5/5/2020	20/7/2020	10/12/2020
Inicio	0	0	20/4/2020	20/4/2020	20/7/2020	20/7/2020
Pruebas individuales de funcionalidad	10	10	20/4/2020	25/4/2020	20/7/2020	25/7/2020
Pruebas sobre la aplicación completa	10	10	25/4/2020	30/4/2020	25/7/2020	30/7/2020
Pruebas en la plataforma web	5	5	30/4/2020	4/5/2020	30/7/2020	4/8/2020
Prueba presencial con alumnos	2	2	4/5/2020	4/5/2020	10/12/2020	10/12/2020
Elaboracion del videotutorial	3	3	4/5/2020	5/5/2020	4/8/2020	5/8/2020
Fin	0	0	5/5/2020	5/5/2020	5/8/2020	5/8/2020

Tabla 4.22: Seguimiento de la fase de pruebas del primer juego

En la fase de analisis del segundo juego vuelven a surgir problemas, ya que se retoma la idea descartada para el primer juego. Esta idea vuelve a ser un problema debido a su dificil adaptación a la gamificación de la asignatura. Pero finalmente y con un retraso considerable se consiguió llegar a un acuerdo en el concepto del segundo juego (tabla 4.23).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	46	82	5/5/2020	20/5/2020	5/8/2020	20/11/2020
Inicio	0	0	5/5/2020	5/5/2020	5/8/2020	5/8/2020
Definición de requisitos del primer juego	2	2	5/5/2020	6/5/2020	5/8/2020	6/8/2020
Proceso creativo para definir el juego	20	50	5/5/2020	15/5/2020	5/8/2020	25/10/2020
Realización del GDD	4	5	5/5/2020	15/5/2020	25/10/2020	28/10/2020
Formación sobre Unity	15	20	10/5/2020	18/5/2020	25/10/2020	10/11/2020
Analisis del modelo de dominio	3	3	15/5/2020	20/5/2020	5/11/2020	20/11/2020
Analisis de casos de uso	2	2	15/5/2020	20/5/2020	5/11/2020	20/11/2020
Fin	0	0	20/5/2020	20/5/2020	20/11/2020	20/11/2020

Tabla 4.23: Seguimiento de la fase de analisis del segundo juego

La fase de diseño del segundo juego no tenía demasiada carga y se ha ajustado a los tiempos previstos (tabla 4.24).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	13	15	20/5/2020	25/5/2020	20/11/2020	25/11/2020
Inicio	0	0	20/5/2020	20/5/2020	20/11/2020	20/11/2020
Investigación de patrones de diseño en Unity	10	12	20/5/2020	24/5/2020	20/11/2020	24/11/2020
Arquitectura del juego	3	3	22/5/2020	25/5/2020	22/11/2020	25/11/2020
Fin	0	0	25/5/2020	25/5/2020	25/11/2020	25/11/2020

Tabla 4.24: Seguimiento de la fase de diseño del segundo juego

La fase de implementación apenas se desvía de los tiempos previstos e incluso gracias a la experiencia adquirida con el primer juego se consigue una implementación más rápida de las funcionalidades (tabla 4.25).

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	48	41	25/5/2020	10/6/2020	25/11/2020	10/12/2020
Inicio	0	0	25/5/2020	25/5/2020	25/11/2020	25/11/2020
Configuración de Unity	2	2	25/5/2020	26/5/2020	25/11/2020	26/11/2020
Obetención y creación de sprites	8	8	25/5/2020	30/5/2020	25/11/2020	30/11/2020
Implementación de las funcionalidades	25	20	25/5/2020	5/6/2020	25/11/2020	5/12/2020
Implementación de la interfaz de usuario	10	7	2/6/2020	7/6/2020	2/12/2020	7/12/2020
Implementación de la conexión con la plataforma web	3	4	5/6/2020	10/6/2020	5/12/2020	10/12/2020
Fin	0	0	10/6/2020	10/6/2020	10/12/2020	10/12/2020

Tabla 4.25: Seguimiento de la fase de implementación del segundo juego

Las pruebas del segundo juego tambien han ido de acuerdo con las estimaciones iniciales y al no ser un proyecto de grán tamaño no ha habido contratiempos (tabla 4.26). La prueba presencial se tuvo que cancelar debido al fin de las clases.

Tarea	Horas estimadas	Horas reales	Inicio estimado	Fin estimado	Inicio real	Fin real
TOTAL	30	30	10/6/2020	25/6/2020	10/12/2020	25/12/2020
Inicio	0	0	10/6/2020	20/6/2020	10/12/2020	20/12/2020
Pruebas individuales de funcionalidad	10	10	10/6/2020	15/6/2020	10/12/2020	15/12/2020
Pruebas sobre la aplicación completa	10	10	15/6/2020	20/6/2020	15/12/2020	10/1/2021
Pruebas en la plataforma web	5	5	20/6/2020	24/5/2020	20/12/2020	12/1/2021
Prueba presencial con alumnos	2	0	24/6/2020	24/5/2020	24/12/2020	14/1/2021
Elaboracion del videotutorial	3	3	24/6/2020	25/5/2020	24/12/2020	15/1/2021
Fin	0	0	25/6/2020	25/6/2020	25/12/2020	15/1/2021

Tabla 4.26: Seguimiento de la fase de pruebas del segundo juego

4.4. Costes

Los costes totales del proyecto han sido de 0⇛. Todo el material usado por el alumno ya estaba amortizado y por tanto no se han generado costes adicionales. Se han usado también herramientas gratuitas.

Por el interés del análisis de costes, se procederá a simular los costes de este proyecto de haberse realizado fuera de un contexto académico.

Se supondrá un salario de programador Junior de 18.751⇛ anuales. Esto sería el un sueldo aproximado de unos 9,62⇛ la hora. Suponiendo un contrato flexible por horas hasta fin de obra, el precio del proyecto sería de 7.138⇛ de inversión en recursos humanos.

Capítulo 5

Minijuego 1: Tipos de Datos

5.1. GDD

5.1.1. Introducción

Se trata de un videojuego didáctico sobre los tipos de datos en java, centrandonos en los tipos básicos. El objetivo es explicar mediante la gamificación los distintos tipos de datos y las diferencias entre ellos así como sus diferentes características. El videojuego consta de dos elementos principales que son las plataformas de datos y las piezas de datos. Cada pieza de datos representa un valor literal y cada plataforma un tipo de dato, la pieza de datos se puede introducir en cada plataforma asignandole un tipo.

El objetivo del juego es asignar una serie de piezas de datos generadas proceduralmente al tipo de dato más correcto haciendo uso de las distintas plataformas de datos. Para hacerlo posible el jugador podrá desplazar horizontalmente la pieza de datos con el fin de introducirla en la plataforma correcta. El juego consta de varios niveles, en cada nivel se van introduciendo distintos tipos de datos.

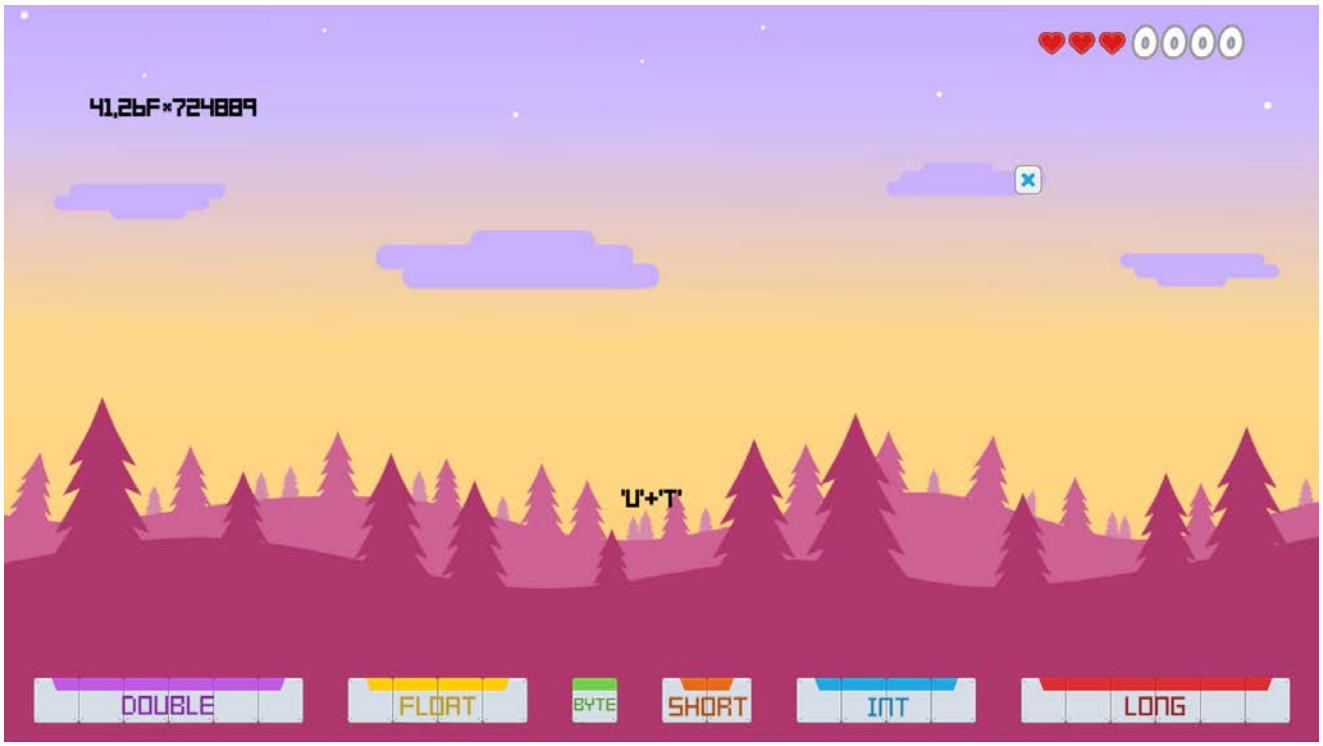

Figura 5.1: Captura de pantalla: Tipos de Datos

En la figura 5.1 podemos ver una imagen del juego en uno de sus niveles. En la parte inferior se pueden observar las plataformas de datos cada una de un color distinto y con el nombre del tipo de dato que representan. En el centro de la pantalla se puede ver la pieza de datos que controla el jugador actualmente. En la parte superior izquierda se muestra la próxima pieza que aparecerá y en la esquina derecha el contador de vidas y la puntuación del jugador.

5.1.2. Audiencia y Plataforma

El desarrollo de este juego serio forma parte de un proyecto de gamificación de la asignatura de Fundamentos de programación, que es impartida en el primer curso de los estudios del Grado de Ingeniería Informática. Por lo tanto la audiencia objetivo son los alumnos de dicha asignatura y los profesores que usarán las estadisticas para comprobar el aprendizaje. La plataforma será una aplicación web que ya ha sido creada anteriormente en otro proyecto.

5.1.3. Mecánicas

Al principio de cada nivel se informará al jugador de los tipos de datos con les que trabajará mediante un cuadro de información, una vez use el botón de comenzar empezará el juego. El elemento más importante es la pieza de datos la cual será el jugador el encargado de mover. Esta pieza siempre se generará en la misma posición y cada vez que sea encajada en un plataforma aparecerá otra que tambien será controlada por el jugador. La pieza es afectada por una fuerza que la hace moverse hacia abajo, dicha fuerza es variable segíún en el nivel y sirve para regular la dificultad.

El jugador podrá realizar dos acciones distintas sobre las piezas:

- Mover horizontalmente la pieza, haciendo uso de las teclas de dirección del teclado o en su defecto haciendo click izquierdo en el lado correspondiente de la pantalla. La velocidad de movimiento dependerá del nivel en el que se encuentre.
- Aumentar la velocidad de descenso de la pieza, haciendo uso de la tecla de dirección del teclado o en su defecto haciendo click izquierdo en la parte inferior de la pantalla.

El segundo elemento más importante serían las plataformas de datos (figura 5.2), las cuales tienen una posicion fija en la parte inferior excepto en el nivel final donde irán subiendo su posición poco a poco con el fin de aumentar la dificultad. Cada plataforma representa un tipo de dato cuyo nombre aparece representado en la misma, tambien serán de un color diferente cada una de ellas. El tamaño será variable en función del tipo de dato que representa y su tamaño de almacenamiento.

Los distintos tipos de datos que hay presentes en el juego son:

- Byte
- Short
- Int
- Long
- Float
- Double

- Char
- String

Los valores generados para cada tipo de dato se corresponderán con los rangos establecidos en java, tambien se hará uso de distintas expresiones con operadares donde el jugador deberá decidir cual es el tipo de dato resultante, algunas expresiones incluirán funciones propias de java.

El jugador contará con un total de 3 vidas para completar cada nivel, para poder superarlo debe colocar todas las piezas sin haber perdido todas las vidas. Se perderá una vida cada vez que una pieza de datos sea colocada en la plataforma de datos incorrecta. El numero de piezas a colocar en cada nivel será distinto. Si el nivel no es superado se le presentará al jugador un cuadro de información a modo de retroalimentación de los errores.

La puntuación se calculará en base al numero de piezas colocadas correctamente y por cada vida perdida se restará una cantidad determinada de puntos, la velocidad tambien será premiada en forma de puntos extras.

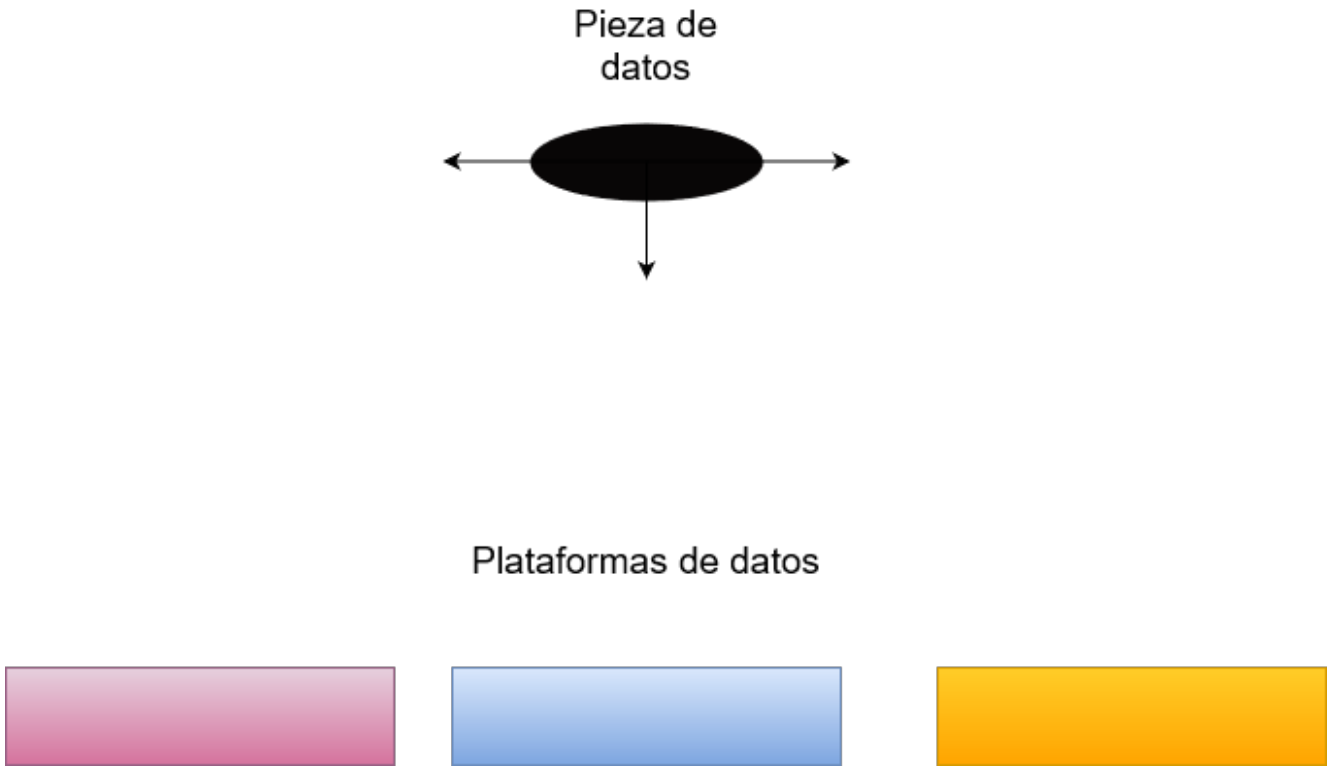

Figura 5.2: Boceto: Tipos de Datos

5.1.4. Niveles

El videojuego constará de 6 niveles, uno de ellos, el primero a modo de tutorial. En cada nivel se irán añadiendo distintos tipos de datos de manera escalonada. Las piezas de datos son generadas de manera dinámica en cada nivel por tanto la repetición de niveles no facilitará su resolución.

Nivel 0
Objetivos de aprendizaje:

- En el nivel tutorial no hay ningún objetivo de aprendizaje.

Objetivos del nivel: Se deberán encajar 10 piezas de colores (azul, roja, verde o amarillo) en cuatro plataformas de colores destintas. La velocidad de caída en este nivel será 3.0 unidades por segundo. En este caso para el tutorial se usarán plataformas de colores azul, verde, roja y amarilla, como los de las piezas. Es necesario no perder las tres vidas para superar el nivel. Servirá para ayudar al jugador a familiarizarse con los controles e introducirle en el juego.

Nivel 1
Objetivos de aprendizaje:

- Distinguir entre valores enteros y decimales.
- Conocer los rangos de los tipos de dato int y double.
- Entender el tipo de dato resultante en expresiones con operadores.
- Reconocer funciones que transforman una valor en un tipo de dato concreto.

Objetivos del nivel: En este nivel la cantidad de piezas de datos a colocar serán 10 con una velocidad de caída de 2.8 unidades por segundo. Los tipos de datos representados con plataformas serán int y double. Es necesario no perder las tres vidas para superar el nivel.

Nivel 2
Objetivos de aprendizaje:

- Ditinguir entre todos los tipos numerícos de java.
- Conocer los rangos de los tipos de dato byte, short, long y float.
- Entender el tipo de dato resultante en expresiones con operadores.
- Aprender el uso de sufijos para denotar distintos tipos de datos.

Objetivos del nivel: En este nivel la cantidad de piezas de datos a colocar serán 10 con una velocidad de caída de 3.1 unidades por segundo. Los tipos de datos representados con plataformas serán byte, short, int, long, float y double. Es necesario no perder las tres vidas para superar el nivel.

Nivel 3
Objetivos de aprendizaje:

- Ditinguir entre todos los tipos numerícos de java, anñadiendo el tipo char.
- Conocer los rangos de los tipos de dato byte, short, long y float.
- Entender el tipo de dato resultante en expresiones con operadores.
- Aprender el uso de sufijos para denotar distintos tipos de datos.
- Aprender el funcionamiento del tipo char en operaciones con enteros.

Objetivos del nivel: En este nivel la cantidad de piezas de datos a colocar serán 10 con una velocidad de caída de 3.45 unidades por segundo. Los tipos de datos representados con plataformas serán char, byte, short, int, long, float y double. Es necesario no perder las tres vidas para superar el nivel.

Nivel 4
Objetivos de aprendizaje:

- Ditinguir entre todos los tipos numerícos de java y los tipos de texto char y string.
- Conocer los rangos de los tipos de dato byte, short, long y float.
- Entender el tipo de dato resultante en expresiones con operadores.
- Aprender el uso de sufijos para denotar distintos tipos de datos.
- Aprender el funcionamiento del tipo char en operaciones con enteros.
- Comprender el funcionamiento de operaciones en cadenas de caracteres.

Objetivos del nivel: En este nivel la cantidad de piezas de datos a colocar serán 10 con una velocidad de caída de 3.45 unidades por segundo. Los tipos de datos representados con plataformas serán char, string, byte, short, int, long, float y double. Es necesario no perder las tres vidas para superar el nivel.

Nivel 5
Objetivos de aprendizaje:

- Ditinguir entre todos los tipos numerícos de java y los tipos de texto char y string.
- Conocer los rangos de los tipos de dato byte, short, long y float.
- Entender el tipo de dato resultante en expresiones con operadores.
- Aprender el uso de sufijos para denotar distintos tipos de datos.
- Aprender el funcionamiento del tipo char en operaciones con enteros.
- Comprender el funcionamiento de operaciones en cadenas de caracteres.

Objetivos del nivel: En este nivel la cantidad de piezas de datos a colocar serán 10 con una velocidad de caída de 4.2 unidades por segundo. Los tipos de datos representados con plataformas serán char, string, byte, short, int, long, float y double. Es necesario no perder las tres vidas para superar el nivel. Adicionalmente en este nivel las plataformas de datos irán elevando su posición con el fin de aumentar la dificultad en el último nivel.

5.1.5. Multimedia

Los elementos multimedia empleados en el desarrollo del juego serán los siguientes:

- **Sprites:** Al tratarse de un juego 2D los elementos del juego serán represantados por sprites, con dos sprites se han creado las plataformas modificando el color para diferenciarlas. Tambien se usarán sprites en la interfaz para representar las vidas y la puntuación así como para la construcción del menú principal.
- **Fuente:** Las piezas de datos serán representadas visualmente con texto, para ellos se han empleado una fuente estilo pixelart que estará presente en todo el juego.
- **Efectos de sonido:** Se usarán diversos efectos de sonidos para retroalimentar al usuarios de los errores y selección de opciones en el menú.
- **Imágenes:** Para cada nivel se usará una imágen de fondo distinto que permitirá diferenciar los niveles entre si.

5.1.6. Versiones

Durante el desarrollo del juego se irán implementado las distintas características del juego de manera escalonado y finalmente añadiendo el contenido en forma de niveles. Estas son las versiones planeadas:

- **Versión 0.5:** En la primera versión unicamente estarán implementadas las mecánicas básicas del juego, haciendo uso de un nivel tutorial en el que se usarán formas de colores en lugar de piezas de datos.
- **Versión 1.0:** Se añade el primer nivel jugable y sus tipos de datos asociados, tambien se incluye el sistema de vidas y una primera versión del menú.
- **Versión 2.0:** Se añaden todos los niveles restantes y se incluyen efectos de sonido. Tambien se añade la retroalimentación de errores.
- **Versión 2.1:** Versión final del juego con el sistema de puntuación y arreglos varios.

5.2. Análisis

5.2.1. Análisis de requisitos

La aplicación tiene que satisfacer ciertos requisitos para poder cumplir con las exigencias del proyecto. Estos requisitos marcarán la solución finalmente elegida para implementar esta aplicación. Se muestran en las tablas 5.1 a 5.13.

RF-01	Carga del menú principal
Descripción	El juego deberá cargar un menú principal, que permitirá iniciar un nivel del juego en concreto, salir del juego y ver el tutorial

Tabla 5.1: RF-01: Carga del menú principal

RF-02	Iniciar nivel
Descripción	El sistema deberá permitir al usuario cargar un nivel

Tabla 5.2: RF-02: Iniciar nivel

RF-03	Salir del nivel
Descripción	El sistema deberá permitir al usuario salir de un nivel

Tabla 5.3: RF-03: Salir del nivel

RF-04	Mostrar ayuda al iniciar nivel
Descripción	El sistema deberá mostrar un panel con ayuda al principio de cada nivel

Tabla 5.4: RF-04: Mostrar ayuda al iniciar nivel

RF-05	Generar pieza de datos
Descripción	El sistema deberá generar datos aleatorios correspondientes al nivel seleccionado

Tabla 5.5: RF-05: Generar pieza de datos

RF-06	Mover pieza de datos
Descripción	El sistema deberá permitir al usuario mover la pieza de datos en el eje horizontal y acelerar su caída hacia abajo

Tabla 5.6: RF-06: Mover pieza de datos

RF-07	Puntuación del jugador
Descripción	El sistema deberá generar una puntuación para cada nivel superado en función del desempeño del jugador

Tabla 5.7: RF-07: Puntuación del jugador

RF-08	Tutorial de inicio
Descripción	El sistema deberá mostrar un tutorial básico explicando los controles y mecánicas principales del juego

Tabla 5.8: RF-08: Tutorial de inicio

RF-09	Cerrar el juego
Descripción	El sistema deberá permitir cerrar el juego guardando los progresos

Tabla 5.9: RF-09: Cerrar el juego

RNF-01	Motor de desarrollo
Descripción	El sistema deberá ser desarrollado haciendo uso del motor de juego Unity

Tabla 5.10: RNF-01: Motor de desarrollo

RNF-02	Lenguaje de programación
Descripción	El sistema deberá ser desarrollado haciendo uso del lenguaje C#

Tabla 5.11: RNF-02: Lenguaje de programación

RNF-03	Plataforma objetivo
Descripción	El sistema deberá ser desarrollado y adaptado para el uso en WebGL

Tabla 5.12: RNF-03: Plataforma objetivo

RNF-04	Comunicación con la plataforma
Descripción	El sistema deberá comunicarse con la plataforma principal del proyecyo cargando y guardando el progreso del jugador así como las estadisticas

Tabla 5.13: RNF-04: Comunicación con la plataforma

5.2.2. Casos de uso

En el digrama de las figuras 5.3 y 5.4 se encuentran los diferentes casos de uso que pueden realizarse en nuestra aplicación. El único actor del sistema será el juegador. Se han dividido los casos de uso en dos escenarios distintos, uno correspondiente a cuando el jugador se encuentra en el menú principal (Figura 5.3) y otro para cuando se encuentra jugando un nivel (Figura 5.4).

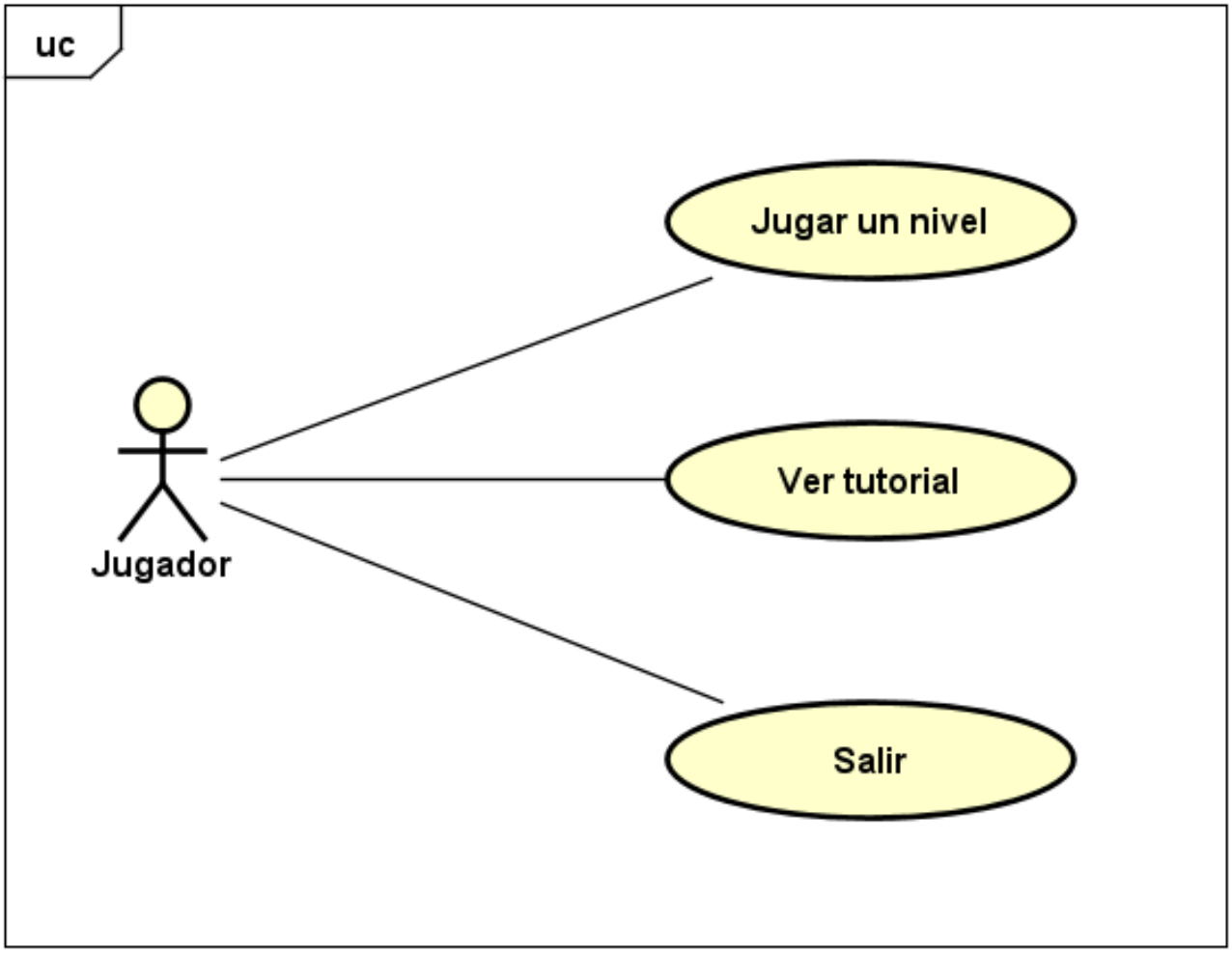

Figura 5.3: Diagrama de casos de uso menú principal: Tipos de datos

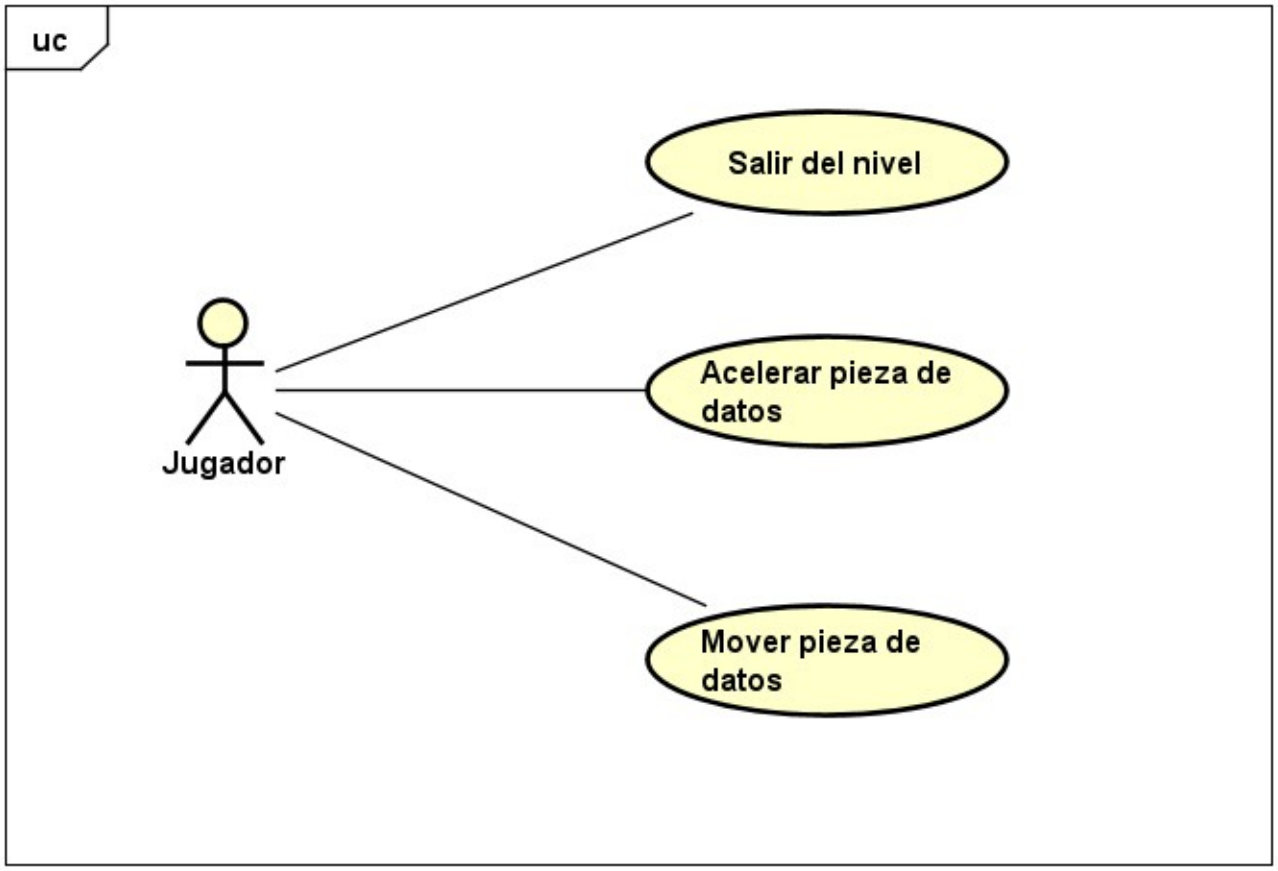

Figura 5.4: Diagrama de casos de uso nivel: Tipos de datos

5.3. Diseño

En este desarrollo se ha usado Unity, el cual nos brinda una librería de clases en la que podemos encontrar recursos para el desarrollo de videojuegos, por lo tanto es fundamental el uso de sus clases para nuestra solución. Debido a esta situación, los diagramas de diseño no se realizarán sobre todas las clases, unicamente sobre los scripts que realicemos que serán descendientos directos de MonoBehaviour. En la figura 5.5 se muestra el modelo de dominio.

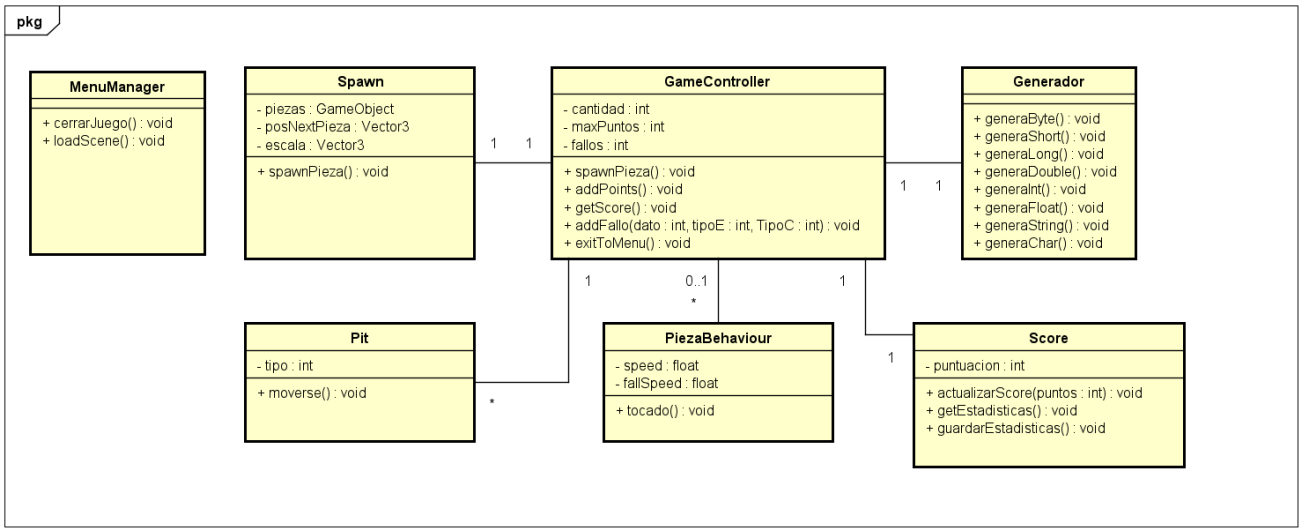

Figura 5.5: Modelo de dominio: Tipos de datos

En esta solución se ha decidido aplicar el siguiente patrón de diseño:

- **Patrón Fachada (figura 5.6):** es un patrón de diseño estructural [2] cuya motivación es la necesidad de estructurar un entorno de programación y reducir su complejidad haciendo que los subsistemas sean más sencillos de ultilizar. Para ello nos proporciona una interfaz unificada que nos permiter minimizar las dependencias entre subsistemas. En nuestra solución hemos utilizado este patrón para la generación de datos dinámicos. Generador es la clase que se encarga de realizar todas las tareas necesarias para generar datos del tipo que se necesite. De esta manera podemos acceder a esta clase de manera sencilla a traves de la fachada.

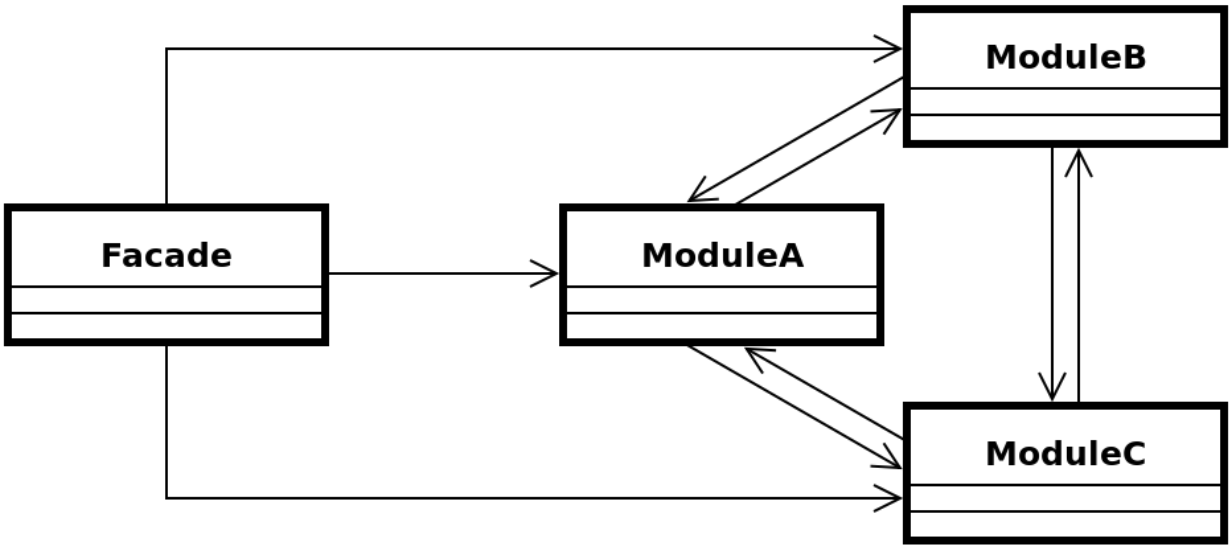

Figura 5.6: Patrón Fachada

5.4. Pruebas

En esta sección se van a mostrar las diferentes pruebas realizadas a lo largo de la implementación de nuestro proyecto.

5.4.1. Pruebas Unitarias

PU-01	Visualizar el videotutorial
Descripción	Al pulsar el botón tutorial del menú principal el juego deberá cambiar de escena y mostrar el tutorial
Resultado	Correcto

Tabla 5.14: PU-01: Visualizar el videotutorial

PU-02	Iniciar un nivel
Descripción	Al pulsar el botón de un nivel del menú principal el juego deberá cambiar de escena e iniciar nivel
Resultado	Correcto

Tabla 5.15: PU-02: Iniciar un nivel

PU-03	Salir del juego
	Al pulsar el botón salir del menú principal el juego deberá cerrarse
Resultado	Correcto

Tabla 5.16: PU-03: Salir del juego

PU-04	Completar nivel
	Al cumplir la condicion de victoria el nivel se marca como superado y se realizan las acciones correspondientes
Resultado	Correcto

Tabla 5.17: PU-04: Completar nivel

PU-05	Fallar nivel
	Al agotar las vidas del nivel, se acaba el juego y se marca como fallado
Resultado	Correcto

Tabla 5.18: PU-05: Fallar nivel

PU-06	Salir del nivel
	Al pulsar el botón de salir, el juego vuelve al menu principal
Resultado	Correcto

Tabla 5.19: PU-06: Salir del nivel

PU-07	Mover pieza horizontalmente
	El jugador puede mover la pieza horizontalmente
Resultado	Correcto

Tabla 5.20: PU-7: Mover pieza horizontalmente

PU-08	Acelerar movimiento descendente de la pieza
	El jugador puede hacer descender la pieza con mayor velocidad
Resultado	Correcto

Tabla 5.21: PU-08: Acelerar movimiento descendente de la pieza

PU-09	Generación de piezas
	Despues de encajar una pieza de datos aparecerá otra que será controlada por el jugador
Resultado	Correcto

Tabla 5.22: PU-09: Generación de piezas

PU-10	Cambio en la puntuación
	Dependiendo de los eventos la puntuación cambia acorde con ellos
Resultado	Correcto

Tabla 5.23: PU-10: Cambio en la puntuación

PU-11	Mostrar vidas correctamente
	El numero de vidas se muestra correctamente y se actualiza con los fallos
Resultado	Correcto

Tabla 5.24: PU-11: Mostrar vidas correctamente

5.4.2. Pruebas Beta

Para esta aplicación se pudieron realizar pruebas beta con alumnos de la asignatura Fundamentos de programación para probar el juego con los usuarios finales.

Para ello se definió un día en el que se realizarón pruebas con unos 20 alumnos, todos ellos cursaban la asignatura que pretendemos gamificar y habían superado el temario necesario para comprender los conceptos que abarca este videojuego.

Antes de comenzar la prueba, se les explico a los alumnos las mecánicas del juego, condiciones de victoria así como los conceptos de la asignatura que pretende explicar. Se realizo una demostracion a modo de tutorial explicando los controles y a continuacion se les proporciono acceso a la plataforma del juego. A lo largo de una sesión de unos 12 minutos pudieron probarlo, cada uno en una máquina independiente, mientras se les resolvian las posibles dudas o problemas que pudiesen surgir en el transcurso de la prueba.

Al finalizar el tiempo de la prueba se realizo una serie de preguntas a los alumnos, entre las cuales estaban conocer cuantos niveles habían logrado completar y su puntuación, la sensaciones que habían tenido sobre el juego asi como su dificultad, posibles mejoras o cambios que tuviesen en mente y ,finalmente, si creían que el videojuego ayudaba a entender los conceptos involucrados.

Tras analizar sus respuestas pudimos encontrar cambios necesarios en el juego que mejorarían la experiencia. A lo largo de las pruebas no se encontró ningún error que arruinase la experiencia pero si cambios para refinar la dificultad, aspecto visual y mecánicas. En general los alumnos quedaron satisfechos con el videojuego considerando que podría resultar útil como material didactico complementario.

Gracias a la retroalimentación de los alumnos se realizaron los siguientes cambios:

- Añadir variedad de fondos para los niveles
- Ajustar la dificultad de los últimos niveles
- Mejoras en la retroalimentacion de errores

Capítulo 6

Minijuego 2: Recursividad

6.1. GDD

6.1.1. Introducción

Se trata de un videojuego didáctico sobre la recursividad. El objetivo es mejorar el aprendizaje mediante la gamificación del funcionamiento de las funciones recursivas y los distintos elementos presentes en cada una de ellas como son la precondición, caso básico y llamada recursiva. El juego consta de un tablero en el que el jugador de debe desplazarse mediante el uso de un dado. De esta manera se ha planteado un juego tipo trivial.

El objetivo del juego es conseguir las 4 estrellas que estan repartidas por el tablero de juego para eso el jugador debe conseguir llegar a la casilla en que se encuentra cada estrella y contestar correctamente a la pregunta de esa casilla. Cada vez que el jugador acierte podrá desplazarse un número de casillas según el número obtenido en el dado. El juego consta de un único nivel en el que la disposición inicial del jugador y las preguntas que se le muestran son totalmente aleatorias.

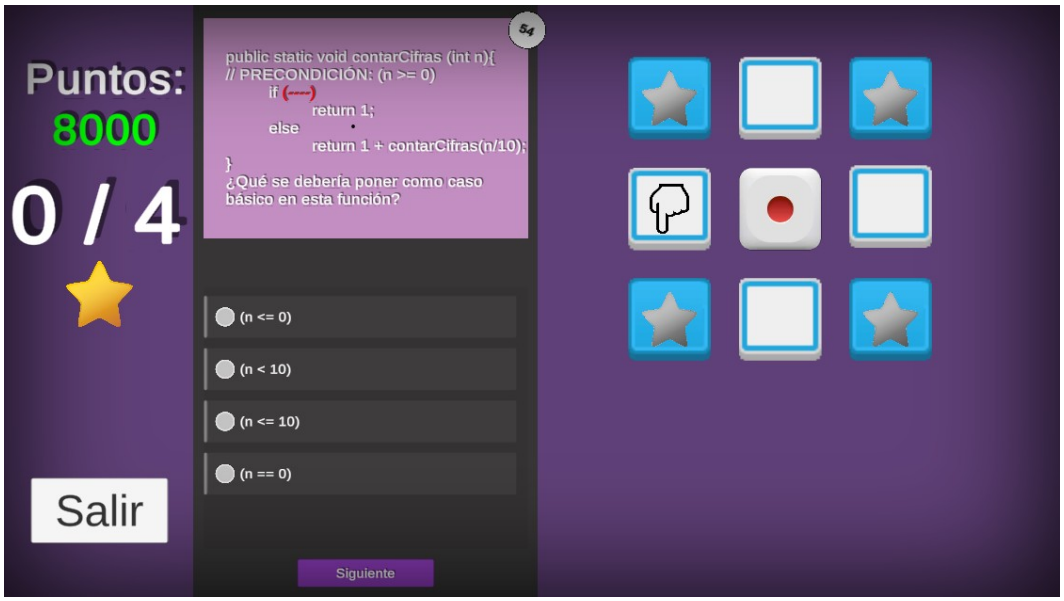

Figura 6.1: Captura de pantalla: Recursividad

En la figura 6.1 podemos ver una imagen del juego en el nivel principal. En la parte izquierda de la pantalla podemos ver los puntos actuales del jugador así como el total de estrellas que ha recogido en esta partida. Tambien existe un botón que permite volver al menu principal. En la parte central podemos ver el cuestionario, en la parte superior se encuentra el enunciado de la pregunta asi como un temporizador que indica el tiempo restante, en la parte inferior se encuentran las distintas respuestas y existe un botón que permite proceder a la siguiente pregunta. En la parte derecha de la pantalla podemos ver el tablero por el cual el jugador debe desplazarse.

6.1.2. Audiencia y Plataforma

El desarrollo de este juego serio forma parte de un proyecto de gamificación de la asignatura de Fundamentos de programación, que es impartida en el primer curso de los estudios del Grado de Ingeniería Informática. Por lo tanto la audiencia objetivo son los alumnos de dicha asignatura y los profesores que usarán las estadisticas para comprobar el aprendizaje. La plataforma será una aplicación web que ya ha sido creada anteriormente en otro proyecto.

6.1.3. Mecánicas

Al principio de cada partida el jugador será posicionado en una casilla aleatoria dentro del tablero y dará comienzo el juego. El elemento más importante del juego es el tablero, que consta de 8 casillas, las cuales pueden ser de dos tipos (figura 6.2):

- **Casilla normal:** esta casilla es de color blanco con el borde azul. Este tipo de casillas no ofrece ninguna recompensa al resolver las preguntas con éxito.
- **Casilla estrella:** esta casilla es de color azul y tiene el icono de una estrella en blanco y negro. Al resolver de manera correcta las preguntas serémos recompensados con una estrella. Al hacerlo el icono cambiará por uno de una estrella en color.

Las 8 casillas formarán un circuito cerrado en el cual se podrá desplazar en ambos sentidos. En el centro del circuito se encontrará un dado que al final de cada turno producirá un valor aleatorio entre 1 y 3 que nos permitirá saber que numero de casillas podemos desplazarnos en el turno actual. El objetivo principal del juego será conseguir las 4 estrellas que estarán repartidas en las distintas casillas con estrella. Para ello, debemos resolver cada casilla correctamente y desplazarnos por el tablero con la ayuda del dado.

El segundo elemento más importante son los cuestionarios de preguntas. Al seleccionar una casilla sobre la que desplazarnos aparecerá un cuestionario con una pregunta y cuatro respuestas. Cada casilla tendrá preguntas diferentes y se seleccionará una de manera aleatoria. Las preguntas consisten en funciones recursivas a las que les falta un trozo, el lugar donde falta será resaltado en color rojo. El jugador debe elegir la solución correcta entre las posibles respuestas. En caso de contestar de manera correcta se lanzará el dado y podrás desplazarte a otra casilla, por el contrario en caso de contestar mal se aplicará una penalización de 1000 puntos y se mostrará otra pregunta de esta casilla.

Pueden existir preguntas tanto de respuesta única como multirespuesta, por defecto existirá un contador de tiempo límite de 60 segundos para contestar la pregunta.

Para poder superar el juego se debe conseguir las 4 estrellas. El jugador tambien contará con una puntuación inicial de 8000 puntos. Para que el juego sea completado con éxito debes mantener al menos la mitad de la puntuación, es decir, 4000 puntos. Por lo tanto se pueden fallar un total de 4 preguntas. Si la puntuación baja de los 4000 puntos el juego se dará por finalizado y el jugador deberá empezar desde el principio.

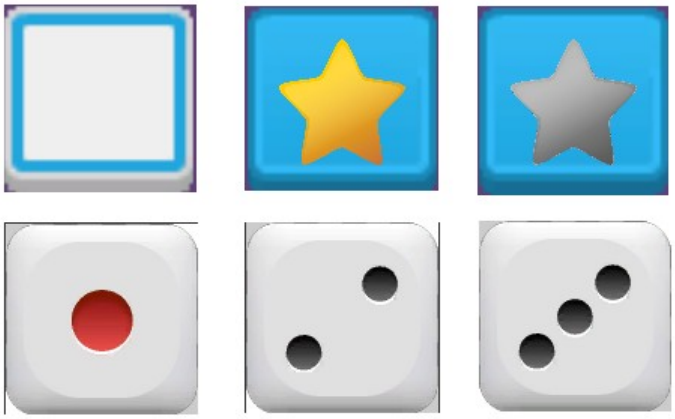

Figura 6.2: Iconos: Recursividad

6.1.4. Niveles

El videojuego no tiene niveles pero para evitar la predecibilidad y que el jugador gane por simple repetición cada casilla tendrá una serie de preguntas distintas entra las que se elige una al azar en cada momento. Para cada pregunta se ha tomado una función recursiva y se han retirado partes tales como la precondición, caso básico o llamada recursiva. Las preguntas están basadas en las siguientes funciones, de las que se han quitado, en cada caso, alguna de sus partes:

Factorial

```
1   public static long factorial (long n){
2   // PRECONDICION: n>=0
3       if (n == 0)
4           return 1;
5       else
6           return n* factorial (n-1);
7   }
8
```

Potencia

```
1   public static int potencia (int a, int b){
2   // PRECONDICION: (a != 0) y (b >= 0)
3       if (b == 0)
4           return 1;
5       else
```

```
6                   return a * potencia(a, b-1);
7               }
8
```

CuentaAtras

```
1               public static void cuentaAtras (int sec){
2               //PRECONDICION:(sec >=0)
3               if (sec == 0)
4               System.out. println ("Fin de cuenta  atrs ");
5               return;
6               else
7               return cuentaAtras(sec-1);
8               }
9
```

Multiplicar

```
1               public static void multiplicar (int a, int b) {
2               // PRECONDICION: (a >= 0) y (b >= 0)
3                   if (b == 0)
4               return 0;
5                   else
6                       return a + multiplicar (a, b-1);
7               }
8
```

Fibonacci

```
1               public static int fibonacci (int n){
2               //PRECONDICION:(n >=0)
3                   if (n == 0)
4               return 0;
5                   else if (n == 1)
6                       return 1;
7                   else
8                       return fibonacci (n-1) + fibonacci(n-2);
9               }
10
```

SumarNumeros

```
1               public static int sumarNumeros (int n){
2               //PRECONDICION:(n >=0)
3                   if (n == 0)
4               return 0;
5                   else
```

```
6                   return n + sumarNumeros(n-1);
7               }
8
```

EsPotencia

```
1               public static boolean esPotencia (int a, int b){
2               // PRECONDICION: (a >= 0) y (b >= 0)
3                   if (a == 0)
4               return false;
5                   else if (a == b)
6                       return true;
7                   else
8                       return esPotencia(a/b, b);
9               }
10
```

ContarCifras

```
1               public static void contarCifras (int n){
2               // PRECONDICION: (n >= 0)
3                   if (n < 10)
4                       return 1;
5                   else
6                       return 1 + contarCifras(n/10);
7               }
8
```

6.1.5. Multimedia

Los elementos multimedia empleados en el desarrollo del juego serán los siguientes:

- **Sprites:** Al tratarse de un juego 2D los elementos del juego serán represantados por sprites, se han usado dos sprites distintos para las casillas así como un sprite de estrella. Para el dado se han usados tres sprites distintos para los tres valores.
- **Música y efectos de sonido:** Se usan efectos de sonido para los aciertos y errores, tambien existe un efecto de sonido para el contador de tiempo. Este juego tambien dispone de una pista de música que suena en bucle durante la partida.
- **Imágenes:** Se usará una imágen de fondo para el menú.

6.1.6. Versiones

Durante el desarrollo del juego se irán implementado las distintas características del juego de manera escalonado y finalmente añadiendo el contenido en forma de niveles. Estas son las versiones planeadas:

- **Versión 0.5:** En la primera versión unicamente estarán implementadas las mecánicas básicas del juego, haciendo uso de cuestionarios con preguntas de prueba.

- **Versión 1.0:** Se añade el primer nivel jugable y algunas preguntas, tambien se incluye el tablero y una primera versión del menú.
- **Versión 2.0:** Se añaden todas las preguntas restantes y se incluyen efectos de sonido. Tambien se añade la retroalimentación de errores.
- **Versión 2.1:** Versión final del juego con el sistema de puntuación y arreglos varios.

6.2. Análisis

6.2.1. Análisis de requisitos

La aplicación tiene que satisfacer ciertos requisitos para poder cumplir con las exigencias del proyecto. Estos requisitos marcarán la solución finalmente elegida para implementar esta aplicación. Se muestran en las tablas 6.1 a 6.13.

RF-01	Carga del menú principal
Descripción	El juego deberá cargar un menú principal, que permitirá iniciar un nivel del juego en concreto, salir del juego y ver el tutorial

Tabla 6.1: RF-01: Carga del menú principal

RF-02	Iniciar nivel
Descripción	El sistema deberá permitir al usuario cargar un nivel

Tabla 6.2: RF-02: Iniciar nivel

RF-03	Salir del nivel
Descripción	El sistema deberá permitir al usuario salir de un nivel

Tabla 6.3: RF-03: Salir del nivel

RF-04	Mostrar el tablero
Descripción	El sistema deberá mostrar el tablero, incluyendo la posición del jugador y el dado.

Tabla 6.4: RF-04: Mostrar el tablero

RF-05	Mostrar cuestionarios
Descripción	El sistema deberá mostrar correctamente las preguntas y respuestas de cada casilla

Tabla 6.5: RF-05: Mostrar cuestionarios

RF-06	Mover pieza en el tablero
Descripción	El sistema deberá permitir al usuario mover su pieza en el tablero

Tabla 6.6: RF-06: Mover pieza de datos

RF-07	Puntuación del jugador
Descripción	El sistema deberá generar una puntuación para cada nivel superado en función del desempeño del jugador

Tabla 6.7: RF-07: Puntuación del jugador

RF-08	Tutorial de inicio
Descripción	El sistema deberá mostrar un tutorial básico explicando los controles y mecánicas principales del juego

Tabla 6.8: RF-08: Tutorial de inicio

RF-09	Cerrar el juego
Descripción	El sistema deberá permitir cerrar el juego guardando los progresos

Tabla 6.9: RF-09: Cerrar el juego

RNF-01	Motor de desarrollo
Descripción	El sistema deberá ser desarrollado haciendo uso del motor de juego Unity

Tabla 6.10: RNF-01: Motor de desarrollo

RNF-02	Lenguaje de programación
Descripción	El sistema deberá ser desarrollado haciendo uso del lenguaje C#

Tabla 6.11: RNF-02: Lenguaje de programación

RNF-03	Plataforma objetivo
Descripción	El sistema deberá ser desarrollado y adaptado para el uso en WebGL

Tabla 6.12: RNF-03: Plataforma objetivo

RNF-04	Comunicación con la plataforma
Descripción	El sistema deberá comunicarse con la plataforma principal del proyecyo cargando y guardando el progreso del jugador así como las estadisticas

Tabla 6.13: RNF-04: Comunicación con la plataforma

6.2.2. Casos de uso

En el digrama de las figuras 6.3 y 6.4 se encuentran los diferentes casos de uso que pueden realizarse en nuestra aplicación. El único actor del sistema será el juegador. Se han dividido los casos de uso en dos escenarios distintos, uno correspondiente a cuando el jugador se encuentra en el menú principal (figura 6.3) y otro para cuando se encuentra jugando (figura 6.4).

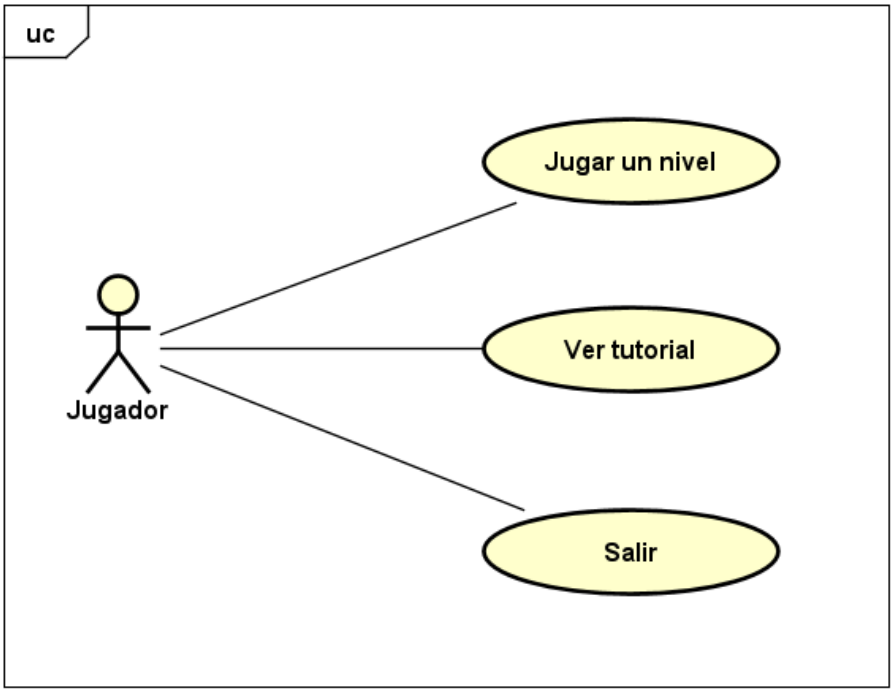

Figura 6.3: Diagrama de casos de uso menú principal: Recursividad

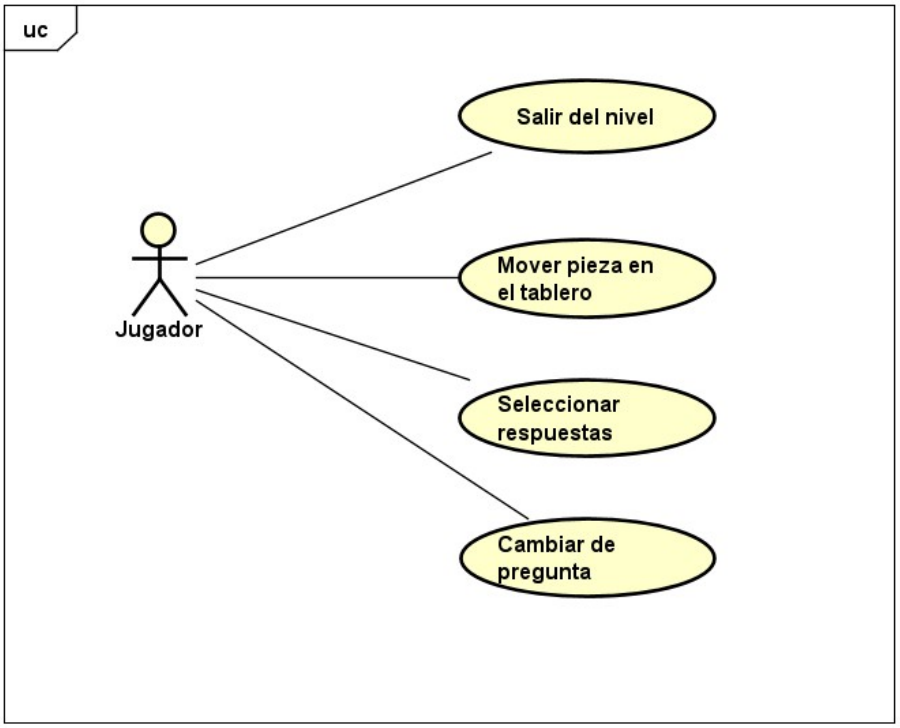

Figura 6.4: Diagrama de casos de uso nivel: Recursividad

6.3. Diseño

En este desarrollo se ha usado Unity, el cual nos brinda una librería de clases en la que podemos encontrar recursos para el desarrollo de videojuegos, por lo tanto es fundamental el uso de sus clases para nuestra solución. Debido a esta situación, los diagramas de diseño no se realizarán sobre todas las clases, unicamente sobre los scripts que realicemos que serán descendientos directos de MonoBehaviour. En la figura 6.5 se muestra el modelo de dominio.

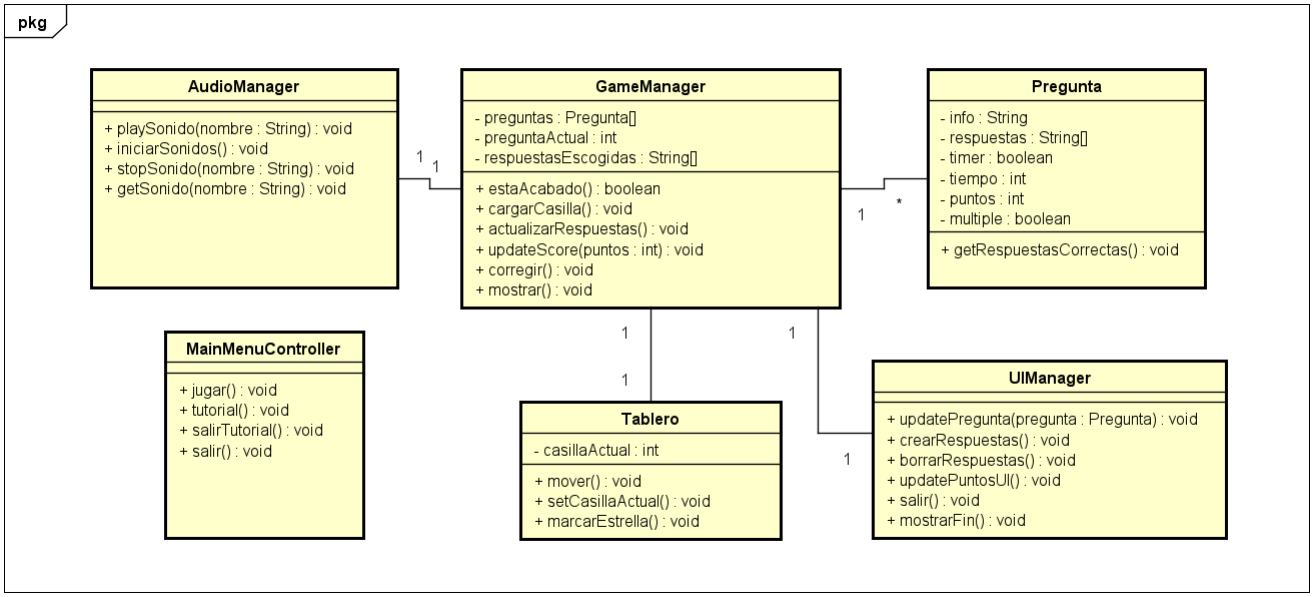

Figura 6.5: Modelo de dominio: Tipos de datos

En esta solución se ha decidido aplicar el siguiente patrón de diseño:

- **Patrón Observador (figura 6.6):** es un patrón de comportamiento [5] que define una dependencia del tipo uno a muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes. Este patrón esta relacionado con algoritmos de funcionamiento y asignación de responsabilidades a clases y objetos. En nuestra solución hemos utilizado este patrón para actualizar los elementos de la interfaz, cada vez que se genera una pregunta o la pantalla de resultado todos los elementos son notificados para realizar su tarea correctamente. En nuestro caso para implementarlo hemos usado el tipo delegado de C#.

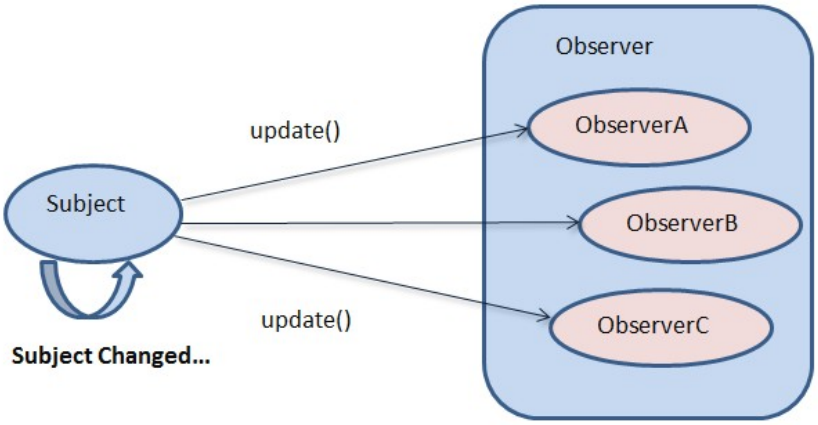

Figura 6.6: Patrón Observador

6.4. Pruebas

En esta sección se van a mostrar las diferentes pruebas realizadas a lo largo de la implementación de nuestro proyecto.

6.4.1. Pruebas Unitarias

PU-01	Visualizar el videotutorial
Descripción	Al pulsar el botón tutorial del menú principal el juego deberá cambiar de escena y mostrar el tutorial
Resultado	Correcto

Tabla 6.14: PU-01: Visualizar el videotutorial

PU-02	Iniciar el nivel
Descripción	Al pulsar el botón de jugar del menú principal el juego deberá cambiar de escena e iniciar nivel
Resultado	Correcto

Tabla 6.15: PU-02: Iniciar un nivel

PU-03	Salir del juego
	Al pulsar el botón salir del menú principal el juego deberá cerrarse
Resultado	Correcto

Tabla 6.16: PU-03: Salir del juego

PU-04	Completar nivel
	Al cumplir la condicion de victoria el nivel se marca como superado y se realizan las acciones correspondientes
Resultado	Correcto

Tabla 6.17: PU-04: Completar nivel

PU-05	Fallar nivel
	Al agotar la puntuación, se acaba el juego y se marca como fallado
Resultado	Correcto

Tabla 6.18: PU-05: Fallar nivel

PU-06	Salir del nivel
	Al pulsar el botón de salir, el juego vuelve al menu principal
Resultado	Correcto

Tabla 6.19: PU-06: Salir del nivel

PU-07	Seleccionar casilla del tablero
	El jugador puede moverse por el tablero
Resultado	Correcto

Tabla 6.20: PU-7: Mover pieza horizontalmente

PU-08	Seleccionar respuesta
	El jugador puede elegir las respuestas entre las que se muestran
Resultado	Correcto

Tabla 6.21: PU-08: Acelerar movimiento descendente de la pieza

PU-09	Corregir pregunta
	Despues de cambiar de pregunta el juego es capaz de corregir la elección del jugador
Resultado	Correcto

Tabla 6.22: PU-09: Generación de piezas

PU-10	Cambio en la puntuación
	Dependiendo de los eventos la puntuación cambia acorde con ellos
Resultado	Correcto

Tabla 6.23: PU-10: Cambio en la puntuación

Capítulo 7

Conclusiones

7.1. Conclusiones

Una vez concluido el desarrollo de este Trabajo de Fin de Grado se han cumplido todos objetivos propuestos al inicio.

Se han construido dos minijuegos independientes que gamifican distintas partes del temario de la asignatura de Fundamentos de Programación del Grado de Ingeniería Informática y que pueden servir de ayuda para al aprendizaje de los conceptos que se exploran en cada juego. Además podría servir de ayuda para cualquier persona que se este iniciando en la programación. Estos juegos han sido integrados dentro de la plataforma principal del proyecto y pueden ser jugados desde ella.

Considero que tanto la metodología aplicada como las tecnologías utilizadas han sido las adecuadas. Destacando sobretodo el uso de Unity, que facilita en gran medida el desarrollo de los videojuegos para pequeños grupos de trabajo y sin el cual este proyecto hubiese sido imposible. El desarrollo iterativo ha sido clave para este proyecto permitiendo hacer frente a los constantes cambios y necesidades de los juegos.

Implicar a los alumnos de nuevas maneras e integrar el aprendizaje con las nuevas tecnologías es un camino a seguir para el futuro de la enseñanza. Puede aportarnos grandes beneficios y poco a poco su presencia irá aumentando en las aulas.

7.2. Trabajo futuro

A lo largo del desarrollo del proyecto han surgido ciertas ideas que se escapaban al alcance del proyecto pero podrían ser implementadas en un futuro para añadir más contenido a los juegos:

- Mejorar el apartado multimedia, creando ”sprites”propios de alta calidad y adecuados al contenido, añadir más y mejores efectos de sonido, animaciones. Pulir el apartado visual cuidando de todos los detalles
- Portar el juego a otras plataformas realizando los ajustes necesarios para su correcto funcionamiento.
- Crear más niveles y añadir nuevas mecánicas que permitan explorar más conceptos de la asignatura.

Referencias

[1] En qué consiste el modelo en cascada. https://openclassrooms.com/en/courses/4309151-gestiona-tu-proyecto-de-desarrollo/4538221-en-que-consiste-el-modelo-en-cascada. [Online; accessed Apr-2021].

[2] Facade (patrón de diseño). https://refactoring.guru/es/design-patterns/facade. [Online; accessed Apr-2021].

[3] Gamificación en el aula: gincana de programación. http://www.aenui.net/ojs/index.php?journal=revision&page=article&op=view&path%5B%5D=402&path%5B%5D=593. [Online; accessed Mar-2021].

[4] Las 5 ventajas principales de la gamificación en el aprendizaje. https://insights.learnlight.com/es/articulos/5-ventajas-gamificacion-aprendizaje/. [Online; accessed Mar-2021].

[5] Observer (patrón de diseño). https://refactoring.guru/es/design-patterns/observer. [Online; accessed Apr-2021].

[6] ¿qué es la gamificación y cuáles son sus objetivos? https://www.educaciontrespuntocero.com/noticias/gamificacion-que-es-objetivos/. [Online; accessed Mar-2021].

[7] ¿qué es unity3d y por qué utilizarlo? https://www.bravent.net/que-es-unity3d-y-por-que-utilizarlo-analizamos-sus-ventajas/. [Online; accessed Apr-2021].

[8] Unity technologies. https://unity.com/es. [Online; accessed Apr-2021].

[9] Tracy fullerton. game design workshop: a playcentric approach to creating innovative games, third edition, Mar 2017.