



Escuela
Politécnica
Superior

Creación de un videojuego para la enseñanza sobre emprendimiento



Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:

Álex Verdú Miralles

Tutor/es:

Pedro Pernías Peco

Junio 2017



Universitat d'Alacant
Universidad de Alicante

Creación de un videojuego para la enseñanza sobre emprendimiento

Autor

Álex Verdú Miralles

Directores

Pedro Pernías Peco

Lenguajes y Sistemas Informáticos



GRADO EN INGENIERÍA MULTIMEDIA



Escuela
Politécnica
Superior



Universitat d'Alacant
Universidad de Alicante

ALICANTE, 7 de junio de 2017

Preámbulo

Los videojuegos han sido parte de mi vida desde que tengo memoria. Empecé jugándolos en una vieja Sega Mega Drive y desde entonces no he dejado de jugarlos en diferentes estilos y plataformas.

Por otro lado considero que el emprendimiento es una forma muy excitante y noble de alcanzar los retos laborales y profesionales con los que cualquier desarrollador de videojuegos podría soñar. Es por ello que decidí unir ambos temas y crear un videojuego dedicado a la enseñanza sobre el emprendimiento.

*A mi pareja, por su apoyo incondicional.
A StackOverflow por estar ahí .
Y a toda la comunidad de desarrolladores que
ofrece su ayuda en internet de forma desinteresada.*

*El hombre de negro huía a través del desierto,
y el pistolero iba en pos de él*

Stephen King.

Índice general

1. Introducción	1
1.1. Emprendimiento y el fenómeno startup	2
1.2. Estado actual del emprendimiento en España	2
1.3. Lean startup	3
1.4. Educación sobre emprendimiento	4
2. Marco Teórico	7
2.1. Actividades en grupo	7
2.2. Juegos de mesa	7
2.3. Videojuegos	9
2.4. Motores de videojuegos	11
3. Objetivos	15
3.1. Objetivo principal	15
3.2. Objetivos específicos	15
3.3. Objetivos secundarios	15
4. Metodología	17
4.1. Producción de contenido	18
4.1.1. Modelos 3D	18
4.1.2. Música y sonidos	18
4.1.3. Motor de videojuego	20
5. Desarrollo	21
5.1. Descripción general	21
5.2. Perspectiva del producto	21
5.3. Funcionalidad del producto	21
5.4. Características de los usuarios	21
5.5. Restricciones	22
5.6. Requisitos específicos	22
5.6.1. Interfaces de usuario	22
5.6.2. Requisitos funcionales	23
5.6.3. Requisitos de usuario	23
5.6.4. Requisitos de interfaz	24
5.6.5. Requisitos de sistema	27
5.7. Requisitos no funcionales	30
5.8. Arquitectura del sistema	30
5.8.1. Arquitectura de Unity3D	30

5.8.2. Arquitectura del juego	37
5.8.3. Storytelling	44
6. Resultados	45
7. Pruebas y validación	49
7.1. Pruebas exploratorias	49
7.2. Pruebas funcionales	49
7.3. Pruebas de usabilidad	49
7.4. Pruebas de compatibilidad	50
8. Conclusiones	51
8.1. Mejoras y ampliaciones	51
8.2. Modelo de negocio	51
Bibliografía	53
A. Anexo I. Documento de diseño de videojuego	55
Visión general	1
Introducción	1
Tema	1
Estilo visual	1
Influencias	1
Menús	2
Menú principal	2
Logros	2
Créditos	3
Opciones	3
Otros juegos	4
Juego	4
Menú conversacional	5
Pausa	5
Personajes	6
Controlables	6
No controlables	6
Mecánicas de juego	8
Movimiento	8
Interacción con personajes	8
Seleccionar personaje	8
Conversar	8
Mecánicas del mundo	9
Cambio de escenario	9
Scrolling paralax	9
Completar objetivo	9

Escenarios	9
Taller de Leonardo	9
Calles de Florencia	9
Palacio de Lorenzo	9
Logia de ingenieros	9
Puerto	10
Guion literario	10
Objetivos	11
Logros	12
Assets requeridos	12
Modelos 3D	12
Sprites	13

Índice de figuras

1.1. Lienzo propuesto por Ash Maurya para el proceso LEAN STARTUP. Fuente: Blog de Javier Megias. http://javiermegias.com/blog/2012/10/lean-canvas-lienzo-de-modelos-de-negocio-para-startups-emprendedores/	3
2.1. Tablero del juego Colonos de Catán. Fuente: Blog del club ERTAI. https://clubertai.wordpress.com/2013/03/10/ludoteca-del-club-los-colonos-de-catan/	8
2.2. Tablero del juego Pandemia. Fuente: Blog Estantería de juegos. http://estanteriadajuegos.blogspot.com.es/2013/12/resena-pandemia.html	9
2.3. Tablero del juego Flea market. Fuente: Tienda online Shopyourway. http://www.shopyourway.com/articles/355984	10
2.4. Captura de pantalla de iPhone con el juego Hipster CEO	12
2.5. Vista del lean canvas en el juego U-startup	12
4.1. Diagrama de Gantt con la distribución temporal y las dependencias de las tareas	19
5.1. Vista del editor de Unity3D donde se pueden ver los componentes de un Gameobject	32
5.2. Diagrama de clases de Unity3D	33
5.3. Bucle de juego del motor Unity3D	36
5.4. Representación del formato utilizado para los ficheros de diálogo	38
5.5. Herramienta inklewriter	39
5.6. Diagrama de flujo nueva conversación	40
5.7. Diagrama de flujo de selección de respuesta	41
5.8. Vista de los escenarios en el inspector de Unity3D	42
5.9. Diagrama de clases del sistema de escenarios	43
6.1. Jugador dialogando con NPC en Da Vinci Startup	45
6.2. Jugador en el escenario Calles de Florencia	46
6.3. Interfaz de usuario durante la partida con los menús desplegados	47
6.4. Menú principal del juego	47
8.1. Lean canvas propuesto para este proyecto	52

Índice de tablas

Índice de Listados

5.1. Código de bucle de juego básico	34
--	----

1. Introducción

El 12 de octubre de 1492 un temerario explorador, Cristobal Colón, y su tripulación pisaron la arena de una isla muy al oeste de Europa conocida como Guanahani. Este hecho marca un hito en la historia de la humanidad pues los cambios culturales, económicos, políticos y militares que produce dan lugar a la llamada Edad Moderna.

Colón vio una oportunidad de negocio en el control de las rutas comerciales que unían Europa con Asia pues eran recorridas por miles de comerciantes que traían especias y productos de lujo desde las tierras de Extremo Oriente. El comercio además se realizaba por tierra, lo que lo convertía en un proceso lento, inseguro e ineficiente, además de enriquecedor para los árabes que controlaban las rutas comerciales.

El proyecto tenía un gran interés económico pues como se ha dicho anteriormente, el control de una ruta comercial con Asia era muy lucrativo, pero a su vez tenía un gran riesgo ya que el futuro de la expedición era tremadamente incierto y había pocas posibilidades de encomendarse al vasto océano y volver para contarlo. Debido a esta incertidumbre sobre el retorno de la inversión a Colón le fue complicado encontrar financiación para su proyecto, hasta que finalmente, tras recurrir a varios monarcas y mecenas, los Reyes Católicos le proveyeron de los recursos necesarios para iniciar su aventura.

Se podría considerar a Cristobal Colón como un emprendedor, a pesar de que el término fue usado por primera vez doscientos años después por el economista Richard Cantillon que define al emprendedor como

La persona que paga un cierto precio para revender un producto a un precio incierto, por ende tomando decisiones acerca de la obtención y el uso de recursos, y admitiendo consecuentemente el riesgo en el emprendimiento.

[Navale, 2013, pág 21].

De esta definición se puede apreciar que un emprendedor inicia proyectos y acepta la incertidumbre y el riesgo que ello conlleva, puesto que en caso de desastre es él quien asume la responsabilidad.

La actitud emprendedora ha sido una constante a lo largo de la historia de la humanidad: desde Cristobal Colón hasta Bill Gates, pasando por Leonardo Da Vinci, Henry Ford o Nikola Tesla; hombres y mujeres con coraje han empezado proyectos bajo una idea prometedora y asumiendo grandes riesgos, motivados por la pasión y las perspectivas de éxito.

El emprendimiento es una actividad especialmente necesaria para el progreso de una sociedad pues es un proceso que crea riqueza, innovación y empleo. Los emprendedores

crean productos y servicios revolucionarios que hacen la vida de las personas más fácil, mejorando por ello su calidad de vida. Además suele ser una opción frecuente en épocas de crisis económicas debido a la escasez de empleo.

1.1. Emprendimiento y el fenómeno startup

Cada vez es más frecuente escuchar el término «startup», pequeñas empresas dedicadas al ámbito tecnológico que alcanzan en pocos años grandes cuotas de mercado y se venden por cantidades astronómicas a empresas más grandes.

El fenómeno goza de tanta popularidad que ha inspirado incluso a series televisivas como «Silicon Valley»¹, que narra las aventuras de un grupo de jóvenes ingenieros que crean una «startup» tecnológica y se enfrentan al reto de sobrevivir en un ecosistema hostil como es el mercado; la película «Piratas de Silicon Valley», que narra la historia de enfrentamiento entre Microsoft y Apple; la película «La red social» que cuenta la historia de Mark Zuckerberg y como crea la red social «Facebook».

Llegado a este punto cabe preguntarse: ¿Qué es exactamente una «startup»? Es un error común pensar que las «startup» son simplemente versiones más pequeñas de empresas grandes. En palabras de los gurús del emprendimiento Steve Blank y Bob Dorf, «Una “startup” es una organización temporal en busca de un modelo de negocio rentable, que pueda repetirse y que es escalable»[Blank and Dorf, 2013].

De la anterior definición se puede extraer que una «startup»:

- Es una organización temporal, es decir, el objetivo no es ser siempre una «startup». El objetivo es convertirse en una empresa consolidada.
- No conoce con seguridad cual va a ser su actividad. En su lugar parten de un modelo de negocio temporal que va evolucionando a medida que interactúa con el mercado.
- Busca un modelo de negocio repetible y escalable, que le permita ejecutar dicho modelo de negocio durante un tiempo indefinido y además expandirse.

El emprendimiento es inherente al fenómeno «startup» pues la incertidumbre es un pilar fundamental al crear una de estas empresas, que ni siquiera tienen un modelo de negocio que se pueda asegurar que va a funcionar.

1.2. Estado actual del emprendimiento en España

Si bien el fenómeno «startup» nació en EEUU y es allí donde está más consolidado, en España es una tendencia igualmente extendida. Atendiendo a cifras de financiación «en 2015, las startups españolas lograron financiación por valor de 500 millones de euros, un 87% más que en 2014, cuando apenas se invirtieron 286 millones de euros» [Fraga, 2016].

¹http://www.imdb.com/title/tt2575988/?ref_=nv_sr_1

Actualmente en nuestro país hay 1783 empresas emergentes distribuidas principalmente en Madrid, Cataluña y la Comunidad valenciana. Dichas empresas se dedican principalmente al ecommerce(22 %), social media(13 %) y las empresas(12 %). En cuanto a la financiación, 172 inversores operan en el ámbito «startup» a lo largo de la península [Startupxplore, 2017] y los fondos que han aportado crecen año a año:

En 2013, tres «startup» lograron rondas de financiación que superaran los 10 millones de euros [...] en 2014, esta cifra aumentó a cuatro [...] el pasado año la explosión no tuvo parangón, ya que hasta 13 startups lograron capitalizar más de 10 millones de euros para fomentar su desarrollo.

[Fraga, 2016].

1.3. Lean startup

«Lean STARTUP» es un modelo de gestión empresarial dinámico ampliamente utilizado en la creación de empresas emergentes. En contraposición a las metodologías tradicionales, «Lean STARTUP» se basa en ciclos de desarrollo cortos que permiten sacar el producto al mercado de forma temprana. De este modo se puede obtener retroalimentación de los clientes en las etapas iniciales de la empresa, lo que da lugar a que el producto cambia y se adapta a las necesidades de los clientes.

El primer paso para crear una «startup» según esta metodología es plasmar las hipótesis sobre el modelo de negocio en el Lean Canvas (ver fig. 1.1)



Lean Canvas is adapted from The Business Model Canvas (<http://www.businessmodelgeneration.com>) and is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported License.

Figura 1.1.: Lienzo propuesto por Ash Maurya para el proceso LEAN STARTUP.
Fuente: Blog de Javier Megías. <http://javiermegias.com/blog/2012/10/lean-canvas-lienzo-de-modelos-de-negocio-para-startups-emprendedores/>

Estas hipótesis no conforman el modelo de negocio definivo, si no que irán evolucionando a lo largo de la vida de la empresa de acuerdo al feedback de los clientes. Esta evolución del producto en relación a los deseos del clientes se denomina **customer development** y es uno de los conceptos claves en Lean startup.

El ciclo de vida de una «startup» se basa en tres pasos fundamentales que se repiten cíclicamente:

- Construir: se diseña el producto en función de las hipótesis que se establecen en el **lean canvas**. En la primera iteración se crea una versión del producto que tenga las mínimas funcionalidades necesarias para aportar valor a los potenciales clientes. Esta versión del producto se denomina **producto mínimo viable**. El objetivo de esta etapa es "comenzar a recopilar datos y medir resultados. Este modelo de producto no busca ser el resultado final sino un producto suficiente para testar la reacción del potencial cliente" [antevenio, 2016].
- Medir: tras contrastar nuestras hipótesis de negocio con los clientes a través del **producto mínimo viable** obtenemos información sobre nuestro producto y sobre la propia empresa mediante **métricas clave**. Dichas métricas (tales como ¿Cuánto cuesta captar un cliente? o ¿Cuánto dinero gastamos mensualmente?) son valoraciones objetivas sobre el rendimiento del producto y de la empresa, y calcularlas de forma periódica es importante ya que permite trazar una evolución y detectar errores y mejoras en la estrategia empresarial.
- Aprender: es una etapa clave ya que si el conocimiento obtenido se aplica, se estará más cerca de crear un producto que los clientes quieran comprar. "este conocimiento adquirido se debe aplicar a un nuevo proceso que comienza de nuevo. Se vuelve a crear un producto, que será una mejora del mismo lo que hace arrancar de nuevo el círculo de crear, medir y aprender" [antevenio, 2016]. Al llegar a este punto las startups se deben plantear si realizar un pequeño ajuste al producto y volver a **iterar** o si bien, en caso de que los resultados del producto hayan sido un desastre, hacer cambios de base al modelo de negocio. Estos cambios que afectan a una o más hipótesis del **lean canvas** se denominan **pivatar** y consisten en "cambiar una hipótesis fundamental sobre el producto, la estrategia, y el motor de crecimiento" [e mooc,].

El proceso se puede realizar cuantas veces sea necesario hasta conseguir el producto que se considere más acorde al cliente. La metodología Lean «startup» no trata de evitar que fallemos en el primer intento de lanzar al mercado nuestro servicio, sino que trata de que ese fallo nos salga más 'barato' al haber empleado una cantidad considerablemente menor de tiempo y de recursos materiales y económicos. [Peláez, 2015] .

1.4. Educación sobre emprendimiento

Dado que la creación de iniciativas empresariales es un fenómeno cada vez más extendido, ha surgido la necesidad de formar a profesionales que sean capaces de poner en

marcha tales proyectos y dirigirlos de forma exitosa. Actualmente existe un gran abanico de opciones formativas, entre las cuales destacaremos los videojuegos. El videojuego, visto como elemento narrativo, supone una poderosa herramienta de comunicación y de experimentación de vivencias.

El tema alrededor del cual gira este Trabajo fin de grado es el videojuego como elemento de aprendizaje sobre emprendimiento. La creación de un videojuego formativo provee a los futuros alumnos no solo de una formación teórica, si no que permite vivir una experiencia inmersiva en una historia que transmita conocimientos importantes a lo largo de la misma. Además es notablemente más divertido aprender jugando a un videojuego que atendiendo a interminables sesiones teóricas. Incluso se puede considerar como formación práctica el jugar ya que el jugador deberá de poner en acción los conocimientos teóricos previamente aprendidos.

2. Marco Teórico

El tema del emprendimiento no solo ha proporcionado ideas para la creación de películas cinematográficas o series de televisión. Como apoyo a los cursos formativos sobre emprendimiento han surgido multitud de juegos para enseñar y aplicar de forma práctica conceptos sobre la creación de iniciativas empresariales.

Estos juegos son una parte importante de la formación ya que permiten aprender de forma más amena. Además el utilizar de forma práctica los conocimientos adquiridos ayuda a que se entiendan mejor y se recuerden durante más tiempo.

Para los juegos mencionados anteriormente existen varios formatos que serán explicados en detalle a continuación.

2.1. Actividades en grupo

Es el tipo más sencillo y tradicional. Solo necesita de los participantes y una actividad previamente elegida. Una de las ventajas que tiene este tipo de actividades es que ponen a las personas en contacto directo, de modo que tienen que dejar a un lado la vergüenza e interactuar como lo harían ante clientes, inversores o trabajadores. Además estos encuentros pueden servir para hacer contactos útiles en un futuro.

Este tipo de actividades potencian las habilidades sociales y la creatividad de los participantes ya que los únicos elementos del juego son las personas, y las mecánicas del juego son sus discursos, explicaciones, gestos, actuaciones, etc.

2.2. Juegos de mesa

Los juegos de mesa mantienen muchos de los aspectos positivos de las actividades en grupo ya que también son presenciales, con las ventajas y desventajas que ello conlleva.

Además estos juegos pueden ser más divertidos debido a que introducen elementos como tableros, cartas, textos o ilustraciones entre otros elementos. Son especialmente interesantes para personas que no se sientan cómodas con las actividades en grupo debido al alto grado de interacción que demandan.

También es más fácil el uso de mecánicas complejas ya que hay instrumentos para contabilizar y describir el estado del juego: dados para contar el número de vidas, poder mágico o turnos; tableros con diferentes casillas, territorios o zonas; fichas de jugador con parámetros, habilidades, y características.

Actualmente hay numerosos juegos de mesa entre los que destacaremos:

Colonos de Catán: ¹ (figura 2.1) Apto para 6 jugadores (a través de una expansión),

¹<http://devir.es/producto/catan/>



Figura 2.1.: Tablero del juego Colonos de Catán. Fuente: Blog del club ERTAI. <https://clubertai.wordpress.com/2013/03/10/ludoteca-del-club-los-colonos-de-catan/>

este juego de gestión de recursos y comercio nos pone en la piel de un colono que debe ir construyendo sus aldeas y caminos. En Colonos de Catán prima tu habilidad para negociar por el contrario y tu capacidad de estrategia a medio y largo plazo. [Entrepreneurship, 2016]

El Catán [...] evita el enfrentamiento tan directo y obliga a negociar para ganar.

Ahí reside una de las claves, en el hecho que de entrada nadie posea recursos suficientes de todos los tipos para progresar. En cada turno se comercia con las materias primas, un trueque básico que permite saber cómo funciona un mercado libre en el que cada uno tiene sus propios intereses. [Albertini, 2015]

Lleva unas 18 millones de copias vendidas [...] Ha aparecido en «The Big Bang Theory» [...] Mark Zuckerberg se ha declarado adicto, «es uno de esos juegos a los que debes jugar si no quieres ser el “margi” de Silicon Valley» [Albertini, 2015]

Pandemia: ² (figura 2.2) En Pandemia somos un grupo de hasta 4 científicos que tienen

²<http://zacatrus.es/pandemia.html>



Figura 2.2.: Tablero del juego Pandemia. Fuente: Blog Estantería de juegos. <http://estanteriadajuegos.blogspot.com.es/2013/12/resena-pandemia.html>

que mantener a raya una serie de virus, o de lo contrario la Humanidad tendrá un grave problema. [...] debes aprender a formar equipo y hacerlo funcionar si quieras tener éxito en tu «startup». En este sentido, Pandemia puede ser la terapia perfecta para ti y tu equipo ya que o colaboráis y funcionáis perfectamente engrasados o correréis el riesgo de fracasar. Y también en la vida real. [Entrepreneurship, 2016]

Flea market:³ (figura 2.3) Juego de dados en el que tendrás que descubrir los tesoros escondidos en un mercado de segunda mano. Tú serás el cliente que compra, y tu objetivo es adquirir bienes lo más barato posible para venderlos por más dinero una vez el dado de la demanda dice que ya son populares de nuevo. [González, 2015]

2.3. Videojuegos

Con el creciente éxito del mercado de los videojuegos y el entretenimiento digital, los videojuegos sobre el emprendimiento no se han hecho esperar. Los hay de diferentes tipos y temáticas aunque todos ellos comparten la esencia de dirigir un negocio.

³<https://boardgamegeek.com/boardgame/172410/flea-market>



Figura 2.3.: Tablero del juego Flea market. Fuente: Tienda online Shopyourway. <http://www.shopyourway.com/articles/355984>

Gestionar un negocio: en este tipo de videojuegos el jugador es el gerente de un negocio como por ejemplo una cafetería o una peluquería. El objetivo es gestionar la actividad diaria del establecimiento y ampliar el mismo utilizando los ingresos obtenidos. Ejemplos de este tipo de videojuegos son «Diner dash⁴» o «My Cafe: Recipes & Stories⁵». Este tipo de juegos tienen un carácter más infantil y recreativo y distan de ser un juego basado realmente en el emprendimiento y el mundo «startup». El motivo es que el peso del juego reside principalmente en la gestión cotidiana del negocio como servir pedidos, cobrar a los clientes o comprar materiales; valores muy importantes para el emprendedor como la creatividad, la estrategia, la pasión o la innovación no suelen tener cabida en estos juegos.

Hipster CEO: ⁶ (figura 2.4) este juego se puede considerar la contraposición al juego mencionado en el punto anterior ya que elimina las tareas de interacción directa con el cliente y en su lugar el jugador se centra en la gestión desde un punto de vista más técnico. El objetivo de este juego es lanzar y dirigir una «startup» con todo lo que ello conlleva.

Entre las tareas del jugador se pueden destacar: contratar personal, dedicar recursos a los diferentes departamentos (ventas, producción o marketing)

Para ello se dispone de un dashboard con varias **métricas clave** sobre la «startup» que ayudan al jugador a tomar decisiones acerca del rumbo de la misma. El juego no dispone de ningún tipo de historia o personajes que controlar, si no que la interacción con el mismo es a través de informes, dashboards, emails, etc.

U-startup: ⁷ (figura 2.5) es un juego desarrollado por la Universidad de Cádiz en España, en colaboración con la Cátedra de Emprendedores y OmnimLab. Lo novedoso de esto es que es el primer videojuego sobre emprendimiento con el que aprenderás a construir tu negocio con la metodología de CANVAS. [del Bosque, 2016]

Este juego se puede considerar la contraposición a los juegos mencionados en el punto anterior ya que explícitamente se muestran elementos típicos de la metodología **lean startup** tales como el **lean canvas**. El juego se basa en completar este canvas con las hipótesis de negocio del jugador a la vez que se completa una aventura en la que un fantasma es el mentor del jugador en el mundo empresarial, la pitonisa ayuda a encontrar los clientes objetivo y otros personajes y escenarios intervienen.

2.4. Motores de videojuegos

La creación de un videojuego sin la utilización de herramientas que agilicen el proceso puede ser una tarea más que ardua. Son muchos los motores de videojuegos que hacen

⁴https://es.wikipedia.org/wiki/Diner_Dash

⁵<https://play.google.com/store/apps/details?id=com.melestacoffeeshop&hl=es>

⁶<http://www.hipsterceo.com/>

⁷<http://ustartup.es/>

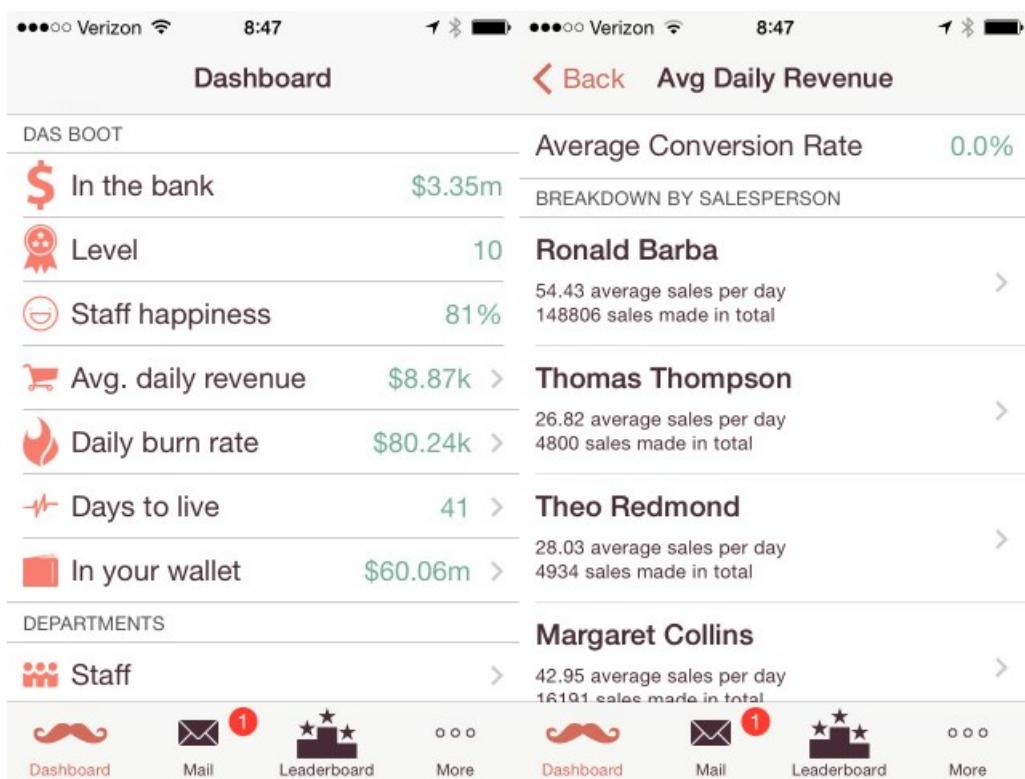


Figura 2.4.: Captura de pantalla de iPhone con el juego Hipster CEO



Figura 2.5.: Vista del lean canvas en el juego U-startup

este proceso más sencillo proporcionando utilidades como sistema de renderizado, motor físico o gestor de sonido. Además estos motores proporcionan código encargado de la gestión de la escena, de la memoria, de las entidades, etc.

3. Objetivos

3.1. Objetivo principal

El objetivo principal de este proyecto la creación un videojuego mediante el cual se puedan aprender conceptos claves acerca del emprendimiento y en concreto sobre la metodología «Lean startup».

3.2. Objetivos específicos

- Emplear el concepto storytelling para diseñar y escribir una historia que se desarrolle a lo largo de conversaciones entre el jugador y los demás personajes.
- Implementar un sistema conversacional en el que el jugador pueda seleccionar las respuestas que desea dar y en función de ello cambie la historia.
- Crear una interfaz de usuario que cumpla con unos criterios de calidad y que permita al jugador interactuar con la aplicación.
- Crear modelos 3D de personajes y otros elementos tales como edificios. Texturizar dichos modelos.

3.3. Objetivos secundarios

- Crear un videojuego usando el motor de videojuegos «Unity3D»¹.
- Planificar el desarrollo de un proyecto de desarrollo de software y llevarlo a cabo.
- Aprender nuevas habilidades sobre el motor de videojuegos «Unity3D» y el lenguaje de programación C#.
- Crear un GDD (documento de diseño de videojuego) donde se documente con precisión como será el juego.
- Aprender nuevas habilidades sobre el software de modelado 3D «Blender»².

¹<https://unity3d.com/es>

²<https://www.blender.org/>

4. Metodología

Para la realización del proyecto, la creación del videojuego, se dispondrán de 6 meses que se dividirán de la siguiente forma: inicialmente un mes y medio para la investigación, especificación del producto y planificación; cuatro meses desarrollando el producto; finalmente medio mes para dar los últimos retoques al juego y lanzar una versión beta.

Para el desarrollo del videojuego se utilizará una metodología ágil, ya que al disponer de poco tiempo para el desarrollo se necesitará tener un producto cuanto antes para poder enfrentarlo al público y obtener el feedback de este. Utilizando una metodología ágil se focaliza el esfuerzo en el desarrollo en lugar de en una excesiva planificación. De esta forma

se trabaja realizando entregas parciales pero funcionales del producto. De ese modo, es posible entregar en el menor intervalo de tiempo posible una versión funcional del producto.

[Martínez, 2014].

Además la utilización de estas metodologías ayuda enormemente a reducir el riesgo: al crear un videojuego, en este caso educativo, a pesar de las mejores intenciones de los desarrolladores no se puede saber con certeza si será del agrado de los jugadores.

Es por ello que la mejor estrategia posible es crear un producto mínimo viable (MVP) y que posteriormente se desarrolle y corrija según los deseos de aquellos que lo jugarán.

Existen actualmente una gran cantidad de metodologías ágiles. Entre las más populares se pueden destacar:

- Scrum: es una metodología orientada a equipos. Proporciona herramientas para el seguimiento diario del proyecto, la planificación de trabajo de forma iterativa y la comunicación y cooperación de los integrantes del grupo.
- Extreme Programming: orientada a equipos con pocos programadores.

se aplica en equipos con muy pocos programadores quienes llevan muy pocos procesos en paralelo. Consiste entonces en diseñar, implementar y programar lo más rápido posible, hasta en casos se recomienda saltar la documentación y los procedimientos tradicionales.

[Pastrana, 2015].

- Kanban: esta metodología propone dividir el trabajo en diferentes etapas bien diferenciadas. El objetivo es evitar los cuellos de botella limitando el trabajo en curso. Para ello, se establece un límite de trabajo en curso, lo que obliga a que cuando una tarea se empieza se debe terminar antes de iniciar una nueva.

La metodología ágil a utilizar será Kanban ya que es tremadamente sencilla de implementar: con unas simples tarjetas se pueden especificar las tareas a realizar, y con un tablero se pueden crear columnas que representan los estados de las diferentes tareas.

Dada la facilidad con la que se puede implementar Kanban, y que no es un sistema directamente orientado a equipos como SCRUM, será muy adecuado para el proyecto.

En cuanto al producto a desarrollar, los cuatro meses de desarrollo se dividirán en iteraciones de duración variable. Al finalizar la segunda iteración se espera tener un mínimo producto viable (MVP) y en las dos siguientes se perfeccionará dicho producto.

En cuanto al tiempo de desarrollo se dividirá en cuatro iteraciones:

- Primera iteración: desarrollo de la parte software del MVP usando placeholders en lugar de los modelos 3D y los demás elementos gráficos
- Segunda iteración: inclusión de los modelos 3D e imágenes definitivas
- Tercera iteración: recolección de feedback y corrección de errores. Optimización del juego
- Cuarta iteración: recolección de feedback y corrección de errores. Detalles finales del juego

Para ilustrar esta distribución de tareas se ha creado un diagrama de Gantt simplificado como se puede apreciar en la siguiente imagen 4.1.

Se utilizará la herramienta online TargetProcess¹ para disponer de un tablero virtual «Kanban» en el cual poner las tarjetas y separarlas por procesos. En este aspecto goza de más popularidad la herramienta Trello², aunque TargetProcess es más completa y ofrece muchas opciones como filtros, métricas y otras utilidades para monitorizar el trabajo.

4.1. Producción de contenido

4.1.1. Modelos 3D

Los modelos 3D a utilizar se obtendrán de sitios web que ofrezcan este tipo de recursos de forma gratuita y con licencia que permita su uso comercial. En caso de que alguno de los modelos no se encontrara se modelaría el mismo utilizando el software Blender.

Se necesitarán modelos para los personajes del juego, los edificios de los escenarios y la decoración de los mismos.

4.1.2. Música y sonidos

Se precisará de una pista de sonido ambiente por cada uno de los escenarios del juego. Se utilizarán canciones que sean coherentes con el espacio histórico en el que transcurre el juego. Para ello se recurrirá a sitios web que ofrezcan piezas musicales gratuitas. La

¹www.targetprocess.com

²<https://trello.com/>

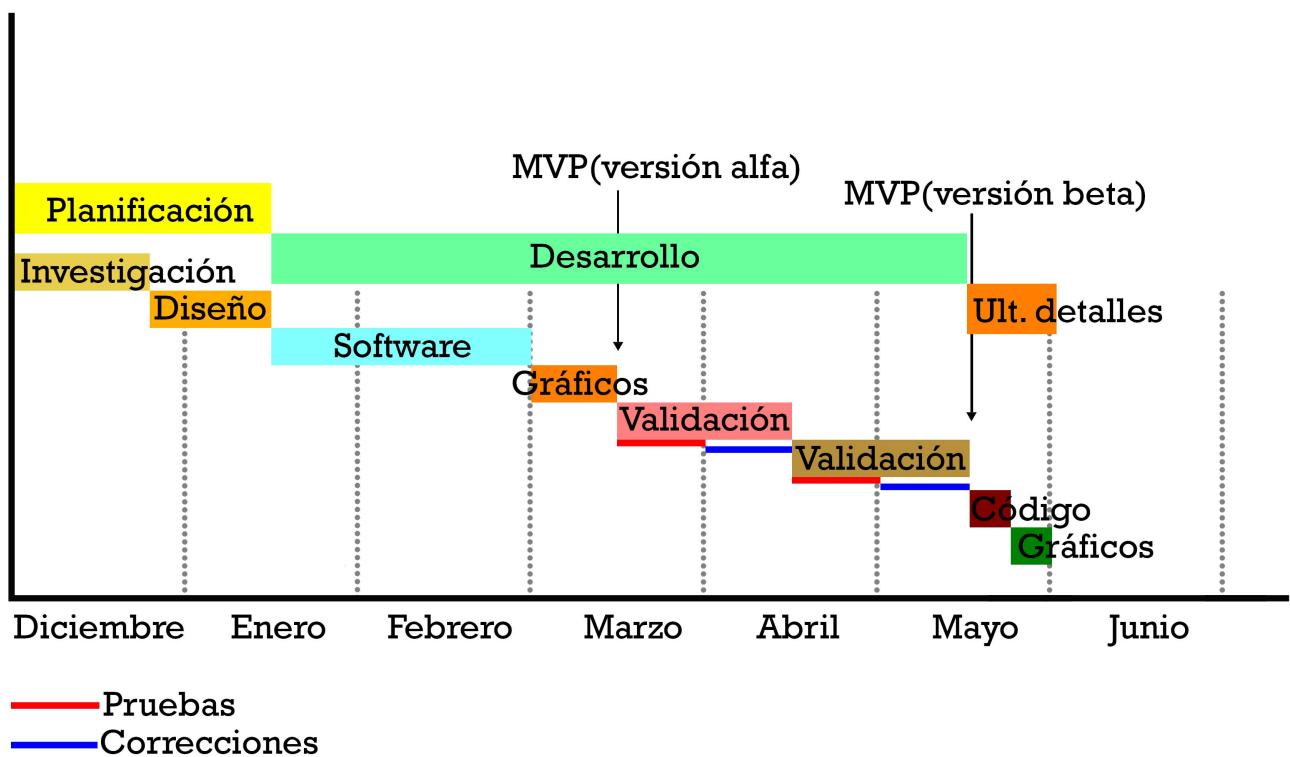


Figura 4.1.: Diagrama de Gantt con la distribución temporal y las dependencias de las tareas

composición de música propia es totalmente inviable por la cantidad de tiempo que consumiría. Si se puede contemplar el uso de Audacity³ para hacer pequeños retoques a las pistas utilizadas.

4.1.3. Motor de videojuego

Como se ha indicado en el capítulo de objetivos se utilizará el motor de videojuegos Unity3D para la creación de este videojuego. El motivo es que actualmente es inviable crear un videojuego sin utilizar ningún tipo de framework, librería o motor.

El uso de un motor de videojuegos agiliza el proceso incluso más que el uso de una librería como puede ser SFML⁴ o Irrlicht⁵. La elección de Unity3D por encima de otros motores como Unreal Engine⁶ se debe a la experiencia previa con el motor Unity3D. Aprender a usar otro motor conllevaría un coste de tiempo que se podría emplear en mejorar el producto.

³<http://www.audacityteam.org/>

⁴<https://www.sfml-dev.org/>

⁵<http://irrlicht.sourceforge.net/>

⁶<https://www.unrealengine.com>

5. Desarrollo

5.1. Descripción general

Para describir el desarrollo se ha decidido seguir el estándar IEEE 830 para la especificación de requisitos ¹.

5.2. Perspectiva del producto

El sistema a construir consistirá en una aplicación móvil para dispositivos Android ² desarrollada con el motor de videojuegos Unity3D ³.

La aplicación no formará parte de un sistema mayor, será un videojuego totalmente independiente. Aun así se hará uso de servicios de terceros tales como librerías de software, frameworks y APIs entre otros.

5.3. Funcionalidad del producto

Resumen de las funcionalidades principales que el producto debe realizar, sin entrar en información de detalle. El videojuego deberá permitir el movimiento del jugador en el plano 2D, además podrá interactuar con los NPCs y dialogar con ellos. Se podrán completar objetivos y logros para de este modo avanzar en la historia.

El juego además enseñará conceptos clave sobre «Lean Startup» y sobre el emprendimiento en general.

5.4. Características de los usuarios

El perfil de un consumidor de formación sobre emprendimiento es muy amplio: desde jóvenes recién graduados llenos de optimismo hasta personas de mediana edad que desean reinventarse y dejar de ser asalariados.

Es por ello que es difícil concretar un perfil ya que son diferentes personas de diferentes edades y perfiles socioculturales las que desean aventurarse en el emprendimiento.

En cualquier caso sí se pueden encontrar aspectos comunes en esta gran variedad de usuarios:

- conocimiento tecnológico y como desenvolverse con aplicaciones móviles

¹www.fdi.ucm.es/profesor/Gmendez/docs/is0809/ieee830.pdf

²www.android.com

³[https://unity3d.com/es/](http://unity3d.com/es/)

- interés por el mundo del emprendimiento y la empresa
- interés por los videojuegos
- interés por los videojuegos educativos

5.5. Restricciones

Existen varias limitaciones a tener en cuenta a la hora de diseñar y desarrollar el sistema:

- para la realización del proyecto se dispondrá de un presupuesto nulo. Es por ello que las herramientas, frameworks y demás productos que se utilicen deberán ser gratuitas.
- el motor de videojuegos a utilizar deberá ser Unity 3D ya que se tiene conocimiento del mismo y no se dispone de tiempo para aprender a utilizar otro motor de videojuegos.
- el lenguaje de programación a utilizar será C# ya que de entre los disponibles para Unity 3D es el más adecuado por potencia, documentación y dominio por parte del desarrollador.
- el sistema operativo objetivo será Android ya junto con los sistemas operativos para ordenador no requiere ninguna licencia para publicar aplicaciones. La plataforma serán los dispositivos móviles ya que es muy sencillo llegar al público de esta plataforma, aunque no tanto hacerse hueco entre dicha audiencia.

5.6. Requisitos específicos

5.6.1. Interfaces de usuario

Respecto a las interfaces de usuario se pueden observar dos estilos claramente diferenciados: las interfaces del menú principal y las de la pantalla de juego. En cuanto a las primeras deberán seguir un estilo minimalista y utilizar controles simples; respecto de las segundas, la simplicidad es obligatoria.

Es un requisito imprescindible que la interfaz mostrada durante el juego no sea intrusiva y entorpezca la experiencia de usuario. Esto se conseguirá disponiendo pequeños botones en la pantalla situados de forma estratégica para que la visión del jugador se centre principalmente en el mundo del juego y los personajes.

Un requisito común de las interfaces de usuario es que debido a que serán mostradas en un dispositivo móvil tendrán que adecuarse a una pantalla pequeña y recibir la interacción del usuario mediante toques en la pantalla del dispositivo.

5.6.2. Requisitos funcionales

Para describir las funcionalidades del sistema se utilizará una aproximación basada en mecánicas. Cada posible acción del usuario sobre el sistema se considerará una mecánica y se definirá como un requisito.

Dichas mecánicas son activadas al detectarse cierto estímulo. Por ejemplo al producir el estímulo de tocar en algún lugar del mundo, se desencadena la mecánica de movimiento.

Al identificador de cada funcionalidad le acompañarán unas siglas (USR, UI, SYS) dependiendo de si dicha funcionalidad se refiere a mecánicas del usuario, la interfaz de usuario o al sistema.

5.6.3. Requisitos de usuario

Identificador	RF-USR-01
Nombre	Mover personaje
Requerimiento	El usuario podrá elegir donde mover al personaje controlado
Descripción	Al tocar con el dedo en cualquier punto de la pantalla el personaje controlado se moverá a esa posición (solo en el eje x)
Prioridad	Imprescindible

Identificador	RF-USR-02
Nombre	Interactuar con NPCs
Requerimiento	El usuario podrá seleccionar un NPC con el que interactuar
Descripción	Al tocar con el dedo sobre un NPC, si se está lo suficientemente cerca se abrirá el menú conversacional
Prioridad	Imprescindible

Identificador	RF-USR-03
Nombre	Cambiar de escenario
Requerimiento	Se podrá navegar entre escenarios
Descripción	Al clicar en una de las puertas de cada escenario se avanzará al escenario asociado a dicha puerta
Prioridad	Imprescindible

5.6.4. Requisitos de interfaz

Identificador	RF-UI-01
Nombre	Navegar menú principal
Requerimiento	Se podrá cambiar entre las diferentes pantallas del menú principal
Descripción	Arrastrando con el dedo en la pantalla hacia la derecha/izquierda se cambiará a la correspondiente pantalla
Prioridad	Imprescindible

Identificador	RF-UI-02
Nombre	Comenzar juego
Requerimiento	Se empezará el juego al seleccionar el botón correspondiente en el menú principal
Descripción	En el menú principal en la vista inicial se podrá comenzar el juego al presionar el botón "play"
Prioridad	Imprescindible

Identificador	RF-UI-03
Nombre	Desactivar/Activar música
Requerimiento	Se podrá desactivar la música ambiente del juego
Descripción	Clicando en el ícono de Música en el menú de opciones se activará/desactivará la música
Prioridad	Baja

Identificador	RF-UI-04
Nombre	Desactivar/Activar sonidos
Requerimiento	Se podrán desactivar los efectos de sonido del juego
Descripción	Clicando en el ícono de Sonidos en el menú de opciones se activarán/desactivarán los efectos de sonido
Prioridad	Baja

Identificador	RF-UI-05
Nombre	Eliminar logros
Requerimiento	Se podrán eliminar los logros conseguidos
Descripción	Clicando en el ícono de Eliminar logros del menú de opciones se eliminarán todos los logros obtenidos
Prioridad	Baja

Identificador	RF-UI-06
Nombre	Ver otros juegos
Requerimiento	Se podrá acceder a la descarga de otros juegos del autor
Descripción	Clicando en los iconos de juegos del apartado "Mas juegos" se accederá a Google Play donde se podrá descargar dicho juego
Prioridad	Baja

Identificador	RF-UI-07
Nombre	Navegar entre logros
Requerimiento	Se podrán seleccionar logros y ver su descripción
Descripción	En el menú de logros se podrá navegar entre los logros usando las flechas de navegación y ver la descripción y la imagen de los mismos
Prioridad	Media

Identificador	RF-UI-08
Nombre	Ver logros desbloquados
Requerimiento	Se podrá ver la cantidad de logros desbloqueados hasta el momento
Descripción	En el menú de logros aparecerá un texto representando el número de logros desbloqueados hasta el momento y el total a conseguir
Prioridad	Baja

Identificador	RF-UI-09
Nombre	Ver descripción de logro
Requerimiento	Se podrá ver una descripción de los logros
Descripción	En el menú de logros se podrá ver un texto descriptivo del logro seleccionado
Prioridad	Media

Identificador	RF-UI-10
Nombre	Navegar entre logros
Requerimiento	Se podrá seleccionar el logro del que se desea ver información
Descripción	En el menú de logros se podrá navegar entre logros tocando y arrastrando en la lista de logros. El logro que se sitúe en el centro de la pantalla será el logro activo
Prioridad	Media

Identificador	RF-UI-11
Nombre	Abrir selector de personaje
Requerimiento	Se podrá abrir un menú donde seleccionar el personaje a controlar
Descripción	Clicando en el ícono del selector de personaje se abrirá una ventana con los iconos de cada personaje. El ícono del personaje actualmente controlado se mostrará en gris y no será seleccionable
Prioridad	Imprescindible

Identificador	RF-UI-12
Nombre	Seleccionar personaje
Requerimiento	Al clicar en un ícono de personaje se cambiará el personaje seleccionado
Descripción	Al clicar en uno de los íconos de personaje se cambia el personaje controlado. En el ícono del selector de personaje aparecerá el ícono del nuevo personaje seleccionado
Prioridad	Imprescindible

Identificador	RF-UI-13
Nombre	Cerrar selector de personaje
Requerimiento	Se podrá cerrar el menú donde seleccionar el personaje a controlar
Descripción	Clicando en el selector de personaje, si este está abierto se cerrará
Prioridad	Imprescindible

Identificador	RF-UI-14
Nombre	Notificar objetivo completado
Requerimiento	Cuando se cumpla el objetivo actual se mostrará una notificación
Descripción	Al completar un objetivo el ícono del indicador de objetivos girará. El texto del nuevo objetivo se añadirá a la lista de objetivos. El objetivo completado se mostrará tachado
Prioridad	Media

Identificador	RF-UI-15
Nombre	Abrir indicador de objetivos
Requerimiento	Se podrá abrir la lista de objetivos
Descripción	Al clicar en el ícono del indicador de objetivos se desplegará una lista con los objetivos completados tachados y el objetivo actual sin tachar
Prioridad	Media

Identificador	RF-UI-16
Nombre	Cerrar indicador de objetivos
Requerimiento	Se podrá cerrar la lista de objetivos
Descripción	Al clicar en el ícono del indicador de objetivos se cerrará la lista si esta estaba abierta
Prioridad	Media

Identificador	RF-UI-17
Nombre	Abrir menú conversacional
Requerimiento	Se abrirá el menú conversacional al interactuar con un personaje
Descripción	Al abrir el menú conversacional se hará zoom sobre los personajes involucrados. Se mostrará un texto en la pantalla correspondiente a lo que dice el NPC y botones con las posibles respuestas
Prioridad	Imprescindible

Identificador	RF-UI-18
Nombre	Contestar NPC
Requerimiento	Se podrá contestar a los NPC usando el menú conversacional
Descripción	Clicando en uno de los botones se contestará al NPC y se actualizará la conversación
Prioridad	Imprescindible

Identificador	RF-UI-19
Nombre	Terminar conversación
Requerimiento	Al alcanzar cierto punto de la conversación se cerrará el menú conversacional
Descripción	Cuando la última decisión tomada tiene asociada la marca correspondiente de fin de conversación el menú conversacional se cerrará. El punto en el que se encuentra la conversación se guarda para retomarla desde ese punto posteriormente
Prioridad	Imprescindible

5.6.5. Requisitos de sistema

Identificador	RF-SYS-01
Nombre	Iniciar menú principal
Requerimiento	Al iniciar el juego el menú principal comienza en la vista correspondiente
Descripción	Al iniciar el juego el menú principal mostrará la vista con el título principal y el botón para iniciar el juego
Prioridad	Imprescindible

Identificador	RF-SYS-02
Nombre	Seleccionar personaje
Requerimiento	Al clicar en un ícono del selector de personaje se cambiará el personaje controlado
Descripción	Cuando se clique en un ícono del selector de personaje se moverá la cámara hasta enfocar al nuevo personaje seleccionado. Cuando se clique en la pantalla se moverá el nuevo personaje controlado
Prioridad	Imprescindible

Identificador	RF-SYS-03
Nombre	Cargar conversación
Requerimiento	Se cargarán las conversaciones desde archivos de texto
Descripción	Al abrir el menú conversacional se cargará el fichero de texto correspondiente a la conversación del personaje. Si ya se ha conversado con ese personaje se cargará la conversación desde el punto guardado
Prioridad	Imprescindible

Identificador	RF-SYS-04
Nombre	Contestar NPC
Requerimiento	Al contestar a un NPC se actualizarán las respuestas y el texto del NPC
Descripción	Al contestar a un NPC se leerá del JSON el nuevo texto y las nuevas respuestas y se actualizará la interfaz para mostrar estos datos
Prioridad	Imprescindible

Identificador	RF-SYS-05
Nombre	Codificar conversaciones
Requerimiento	Las conversaciones estarán codificadas en formato JSON y guardadas en ficheros de texto
Descripción	Cada NPC tendrá un fichero de texto asociado con la conversación que ofrece en formato JSON. Los elementos del JSON serán nodos con texto y nodos hijos con las respuestas asociadas a ese texto. Las respuestas podrán tener "markups." asociadas que indican eventos
Prioridad	Imprescindible

Identificador	RF-SYS-06
Nombre	Completar objetivo
Requerimiento	Se completarán objetivos al tomar decisiones con la markup correspondiente
Descripción	Cuando se elige una opción de conversación con la "markup" de objetivo asociada, se completará dicho objetivo
Prioridad	Imprescindible

Identificador	RF-SYS-07
Nombre	Cambiar de escenario
Requerimiento	Cuando el jugador cambia de escenario se modifica la escena
Descripción	Cuando se produzca un cambio de escenario se desactivará de la escena el escenario abandonado y el jugador aparecerá en el punto de "spawn" asociado al nuevo escenario
Prioridad	Imprescindible

Identificador	RF-SYS-08
Nombre	Guardar juego
Requerimiento	El estado del juego se guardará al salir
Descripción	Cuando el usuario salga de la aplicación o al menú principal, el estado del juego se guardará en la memoria del dispositivo
Prioridad	Baja

Identificador	RF-SYS-09
Nombre	Cargar juego
Requerimiento	Al iniciar el juego se cargarán las partidas guardadas
Descripción	Cuando se inice el juego se cargarán las partidas guardadas en caso de existan. De ser así, al iniciar la partida el mundo se encontrará en el estado de la partida guardada
Prioridad	Baja

Identificador	RF-SYS-10
Nombre	Actualizar conversaciones
Requerimiento	Al completar un objetivo se actualizan las conversaciones
Descripción	Al completar un objetivo en una conversación, todas las demás conversaciones se actualizarán a un punto correspondiente de la misma para reflejar el objetivo completado. Las nuevas conversaciones empezarán desde dicho punto.
Prioridad	Imprescindible

5.7. Requisitos no funcionales

- Rendimiento: el sistema debe de proporcionar una experiencia de juego óptima. Para ello el flujo de juego debe de ser fluido, sin parones o interrupciones.
- Usabilidad: el sistema debe de ser sencillo de utilizar. De modo que personas sin un contacto o entrenamiento previo sepan desenvolverse por el sistema
- Costo: el coste del sistema deberá ser cero

5.8. Arquitectura del sistema

Para poder explicar la arquitectura del sistema construido es necesario en primer lugar exponer la arquitectura del motor de videojuegos utilizado. El motivo es que el sistema construido tendrá que cumplir con las normas y reglas del motor, ya que se usarán las herramientas que este provee. Es por ello que se puede afirmar que la arquitectura del videojuego está condicionada y subordinada a la del motor utilizado.

Dado que para la realización de este proyecto se ha utilizado el motor de videojuegos Unity3D⁴, se explicará a continuación el diseño de dicho motor.

5.8.1. Arquitectura de Unity3D

Unity3D basa su diseño en el modelo entidad-componente⁵. En este diseño, las entidades del mundo del juego obtienen su funcionalidad mediante la agregación de diferentes componentes.

En el paradigma de la programación orientada a objetos⁶ el comportamiento de una entidad se define en la clase que la representa. Sin embargo, para reducir la duplicación de código se emplea el mecanismo de la herencia, por el cual una clase puede heredar de otra y de esta forma obtener su comportamiento y ampliarlo. El problema que reside en basar el diseño en jerarquías de herencia es que a medida que crece la complejidad del sistema crece el "árbol" que forman las clases. Llegados a este punto es probable que se de el caso en el que se precisa cambiar la funcionalidad de una de las clases, y que al hacerlo se modifique de forma involuntaria el comportamiento de otras clases que heredan de la modificada. Llevando el caso a un punto más extremo, se podría dar la situación en la que el comportamiento de una clase sea incompatible con el comportamiento heredado de una clase padre.

Para solucionar este problema el diseño entidad-componente propone crear entidades que actúen como meros contenedores. Dichos contenedores agregarán componentes que les darán la funcionalidad que los definirá. Por ejemplo tanto un jugador como los objetos del escenario tendrían el componente Mesh (malla gráfica) pero solo el jugador tendría el componente Movement (movimiento). Como se puede observar este diseño es altamente

⁴<https://unity3d.com/es/>

⁵<https://www.genbeta.com/programacion-de-videojuegos/diseño-de-videojuegos-orientado-a-entidades-y-componentes>

⁶<https://desarrolloweb.com/articulos/499.php>

flexible, ya que si se toma la precaución de hacer los componentes lo suficientemente genéricos, se pueden aplicar a varias entidades, de forma que hay una gran reutilización de código.

Otra ventaja es que se produce un muy bajo acoplamiento ya que las funcionalidades están encapsuladas en los componentes y estos son independientes entre sí.

La forma en la que Unity3D aplica este diseño entidad-componente es mediante el uso de «Gameobjects» (entidades) y components. Los «Gameobjects» son la entidad fundamental en Unity3D: todos los elementos que existan en la escena son un «Gameobject». Además todas las entidades en Unity3D (en adelante Gameobjects) incorporan como mínimo un componente. Dicho componente se llama «Transform» y se encarga de la gestión de la posición, la rotación y la escala del «Gameobject» que lo posee.

En la siguiente imagen (ver fig. 5.1) se puede observar los componentes que posee el «Gameobject» Leonardo.

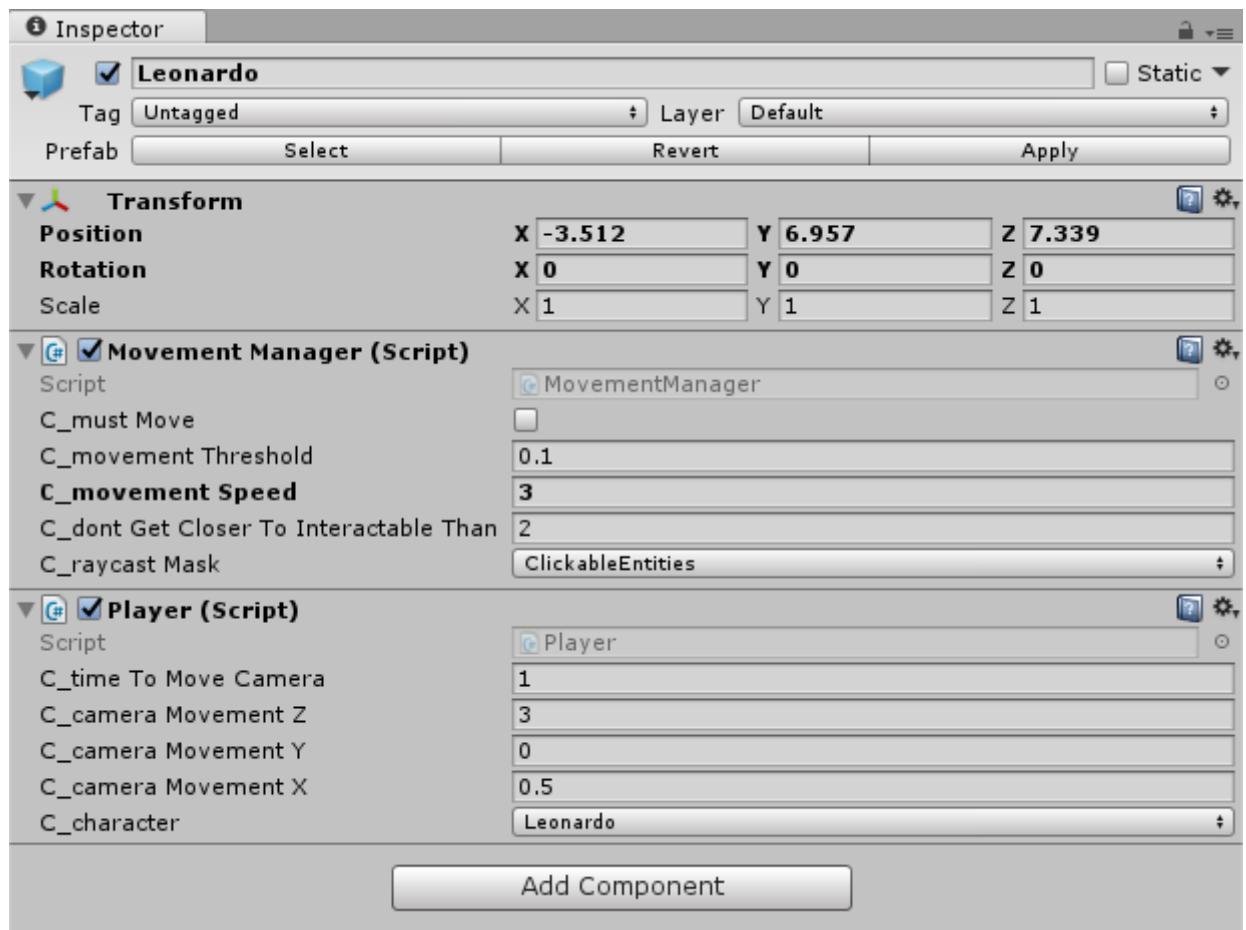


Figura 5.1.: Vista del editor de Unity3D donde se pueden ver los componentes de un Gameobject

Como se ha mencionado anteriormente el componente «Transform» es obligatorio en todos «Gameobjects». También se pueden ver otros dos componentes, «Movement Manager» y «Player», que han sido creados por el desarrollador. Dichos componentes dotan a la entidad, que en este caso es el personaje controlable por el jugador, de las características que lo hacen único. Entre otras cosas le dan la habilidad de reaccionar ante los inputs del usuario y moverse por el escenario. También se pueden observar una serie de variables con parámetros. Dado que cada instancia de los componentes es única para la entidad que lo agrega, se pueden personalizar para que incluso entre entidades con los mismos componentes se comporten de manera diferente.

En la siguiente imagen (ver fig. 5.2) se puede observar un diagrama de clases que muestra la estructura que compone el sistema de componentes y los Gameobjects.

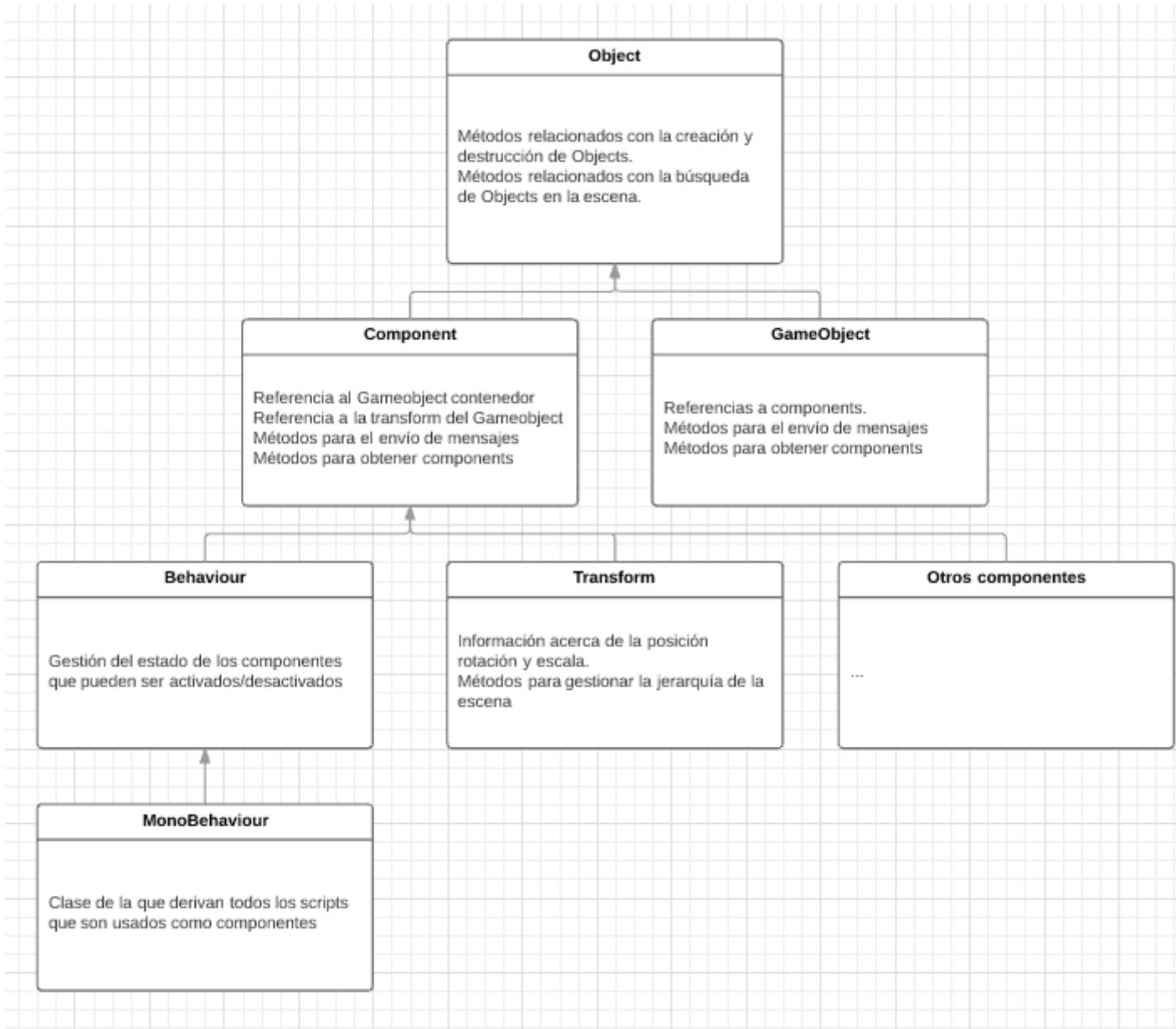


Figura 5.2.: Diagrama de clases de Unity3D

Los componentes que provee Unity3D derivan de la clase Component. En el caso de la imagen solo aparece el componente «Transform», pero hay muchos más como «Renderer», «Rigidbody», «Collider», etc. En la clase especializada cada componente contendrá la lógica necesaria para desempeñar su función, y el la clase Component se referenciará

al «GameObject» que lo contiene. Cabe notar que un componente no puede existir de forma independiente. Debe estar siempre asociado a una entidad.

La clase MonoBehaviour es también muy importante ya que es la clase base para los scripts creados por los desarrolladores. Cualquier fragmento de código que se desee añadir como componente a una entidad deberá heredar de dicha clase.

Por supuesto se pueden escribir scripts que no hereden de la clase MonoBehaviour, pero en ese caso no podrán ser asignados como componentes a una entidad. Sin embargo no dejan de ser útiles ya que pueden ser usados como POCO⁷, definir interfaces, contener lógica o cualquier otra cosa que el desarrollador deseé.

En última instancia los scripts que no heredan de MonoBehaviour deben de ser utilizados desde un script que sí lo haga, ya que este es el único punto de entrada que tienen hacia el sistema de Unity3D.

El último aspecto de la arquitectura que se comentará es el bucle de juego: una parte fundamental de cualquier videojuego. El bucle de juego es la parte software más importante en cualquier videojuego. En él se ejecutan las tareas que hacen que el juego 'esté vivo'. En el siguiente fragmento de código (ver fig. 5.1) se presenta un ejemplo de bucle de juego básico:

Listado 5.1: Código de bucle de juego básico

```
while(true)
{
    ProcesarInput();
    ActualizarJuego();
    Renderizar();
}
```

En este fragmento de código se pueden observar las tres tareas fundamentales de las que se compone cualquier videojuego:

- Procesar input: consiste en capturar la interacción del usuario con el sistema mediante el hardware. En esta etapa se puede aplicar algún tipo de filtrado sobre dicho input.
- Actualizar juego: se actualiza el estado del juego. Esto consiste en actualizar IA, realizar las acciones del personaje, actualizar el mundo, efectos de objetos, etc.
- Renderizar: los elementos del juego se procesan para ser dibujados en la pantalla.

El ejemplo de bucle mostrado anteriormente es tremadamente básico pero sirve para explicar las bases de un bucle de juego. Probablemente ningún videojuego moderno lo utilice ya que tiene múltiples inconvenientes como que la frecuencia de actualización del juego dependerá de la carga de trabajo y del rendimiento del sistema que la procese. Actualmente existen variaciones del bucle que solucionan este problema, pero son temas que quedan fuera del alcance de este proyecto.

⁷https://es.wikipedia.org/wiki/Plain_Old_CLR_Object

En la siguiente imagen (ver fig. 5.3) se puede observar el bucle de juego utilizado por el motor Unity3D, y evidentemente por el juego creado para este proyecto:

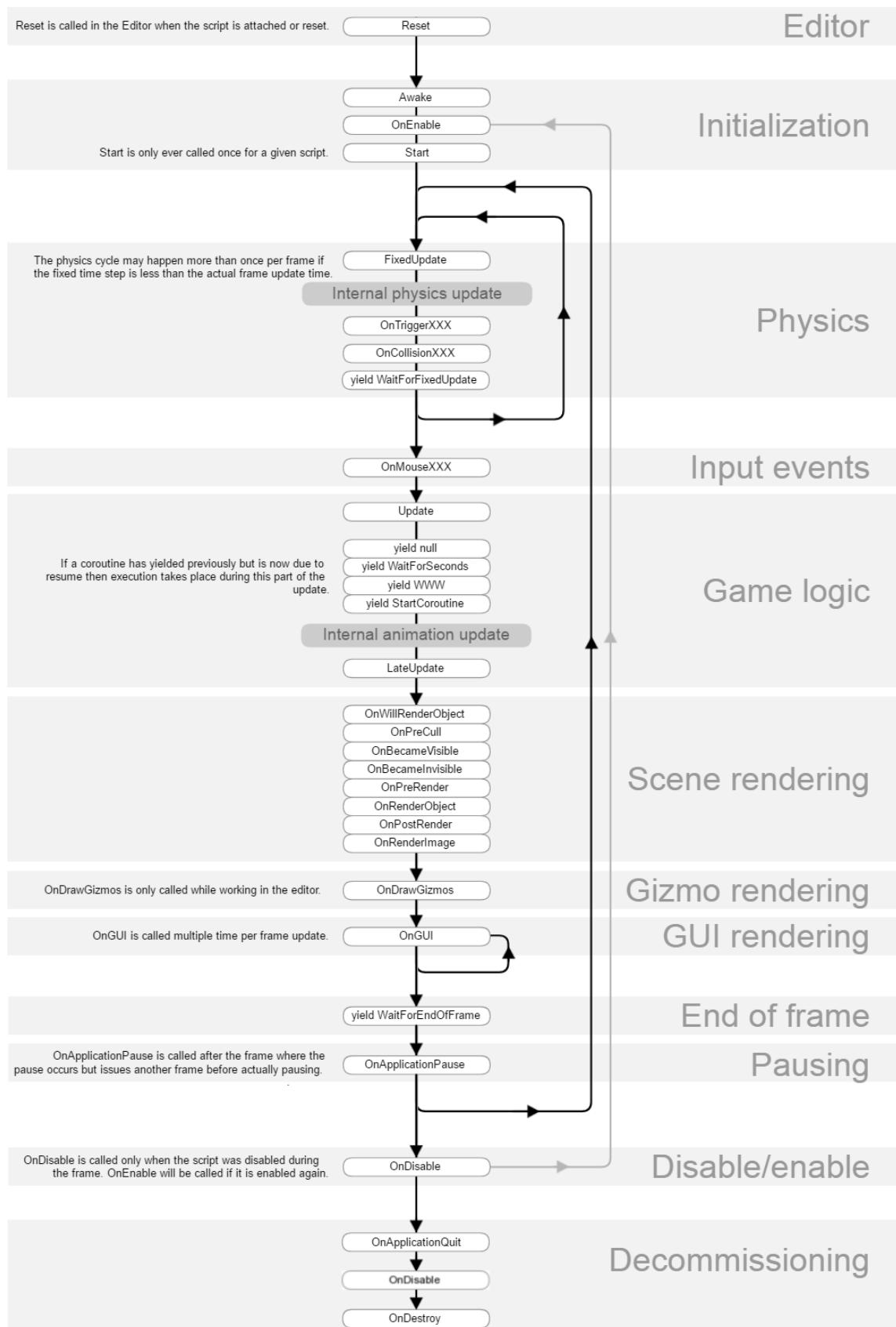


Figura 5.3.: Bucle de juego del motor Unity3D

En la imagen se pueden observar las 3 fases clave del bucle de juego: la captura de acciones del jugador, que se produce en la parte denominada como «Input Events»; la actualización del mundo de juego que se lleva a cabo en «Physics» y «Game logic»; finalmente el dibujado del mundo, que se produce en «Scene rendering», «Gizmo rendering» y «GUI rendering».

5.8.2. Arquitectura del juego

A continuación se describirá la arquitectura utilizada para desarrollar los componentes más importantes del juego. Estos son: el sistema de conversaciones y el sistema de escenarios.

Sistema de conversaciones

El sistema de conversaciones es el más importante del juego ya que los diálogos con los personajes del juego son el hilo conductor de este.

Los diálogos en lugar de estar insertados directamente en el código se almacenan en ficheros de texto plano independientes. Esto permite que se puedan modificar los diálogos sin tener que recompilar el código del juego. Además permite que cualquier persona sin conocimientos de programación escriba diálogos.

Cada diálogo es un archivo que sigue el formato JSON⁸. Las conversaciones tienen una estructura arbórea donde los nodos representan la parte del diálogo correspondiente al NPC y las ramas salientes de dicho nodo representan las posibles contestaciones. La representación de una conversación en el formato JSON sigue la estructura que se puede ver en la siguiente imagen (ver fig. 5.4).

⁸<https://geekytheory.com/json-i-que-es-y-para-que-sirve-json>

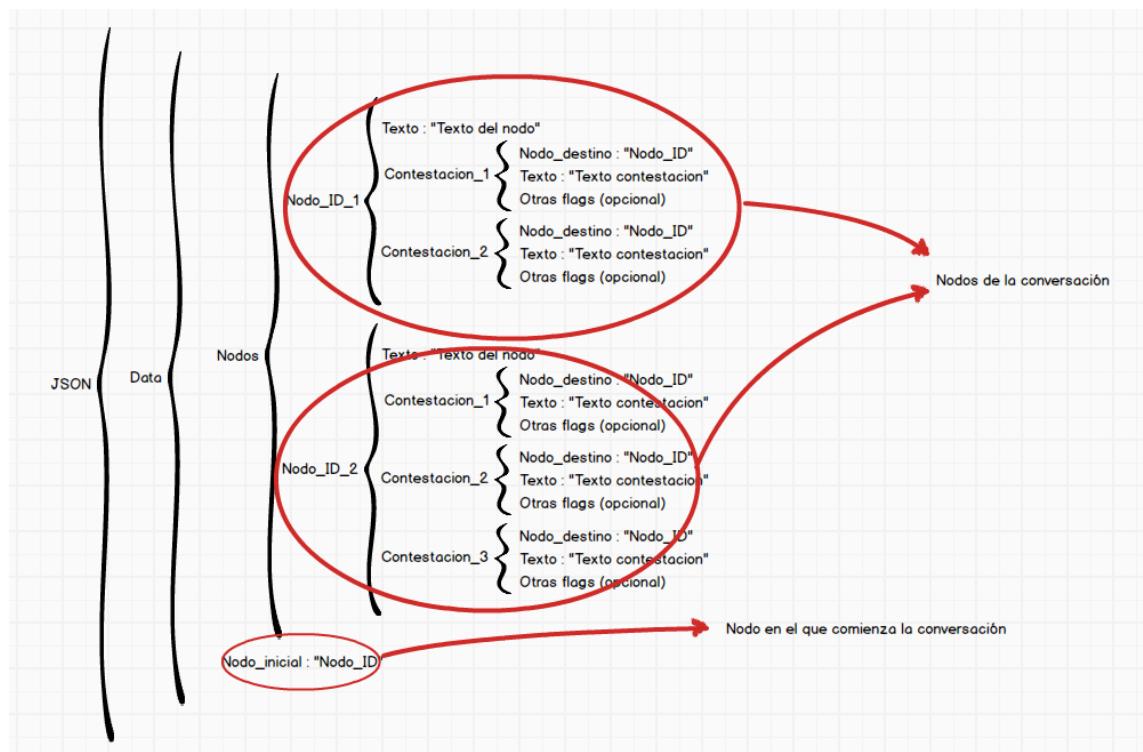


Figura 5.4.: Representación del formato utilizado para los ficheros de diálogo

Como se puede observar, los datos contenidos en el JSON son un array de objetos Nodo y un string que indica el nodo inicial de la conversación.

Los objetos Nodos están identificados por una cadena de texto que se compone de las tres primeras palabras del diálogo. Cada Nodo contiene el texto del NPC y las posibles contestaciones que el jugador le puede dar.

Las contestaciones se componen de el identificador del nodo al que conduce dicha contestación y el texto de la contestación en sí. Además puede contener ,o no, diferentes «flags» que le dan a la elección una funcionalidad extra. Entre las «flags» disponibles se encuentran la de acabar la conversación, desbloquear un objetivo o desbloquear un logro.

Para la creación de los ficheros de diálogos se puede escribir el JSON manualmente rellenando con los datos deseados o se puede utilizar la herramienta inklewriter⁹. Dicha herramienta presenta una forma más sencilla de escribir conversaciones de múltiple respuesta (ver fig. 5.5). Además cuenta con la posibilidad de exportar el trabajo final a un fichero de texto en formato JSON. El parser JSON del videojuego está construido específicamente para poder procesar los diálogos construidos con inklewriter.

⁹<http://www.inklestudios.com/inklewriter/>

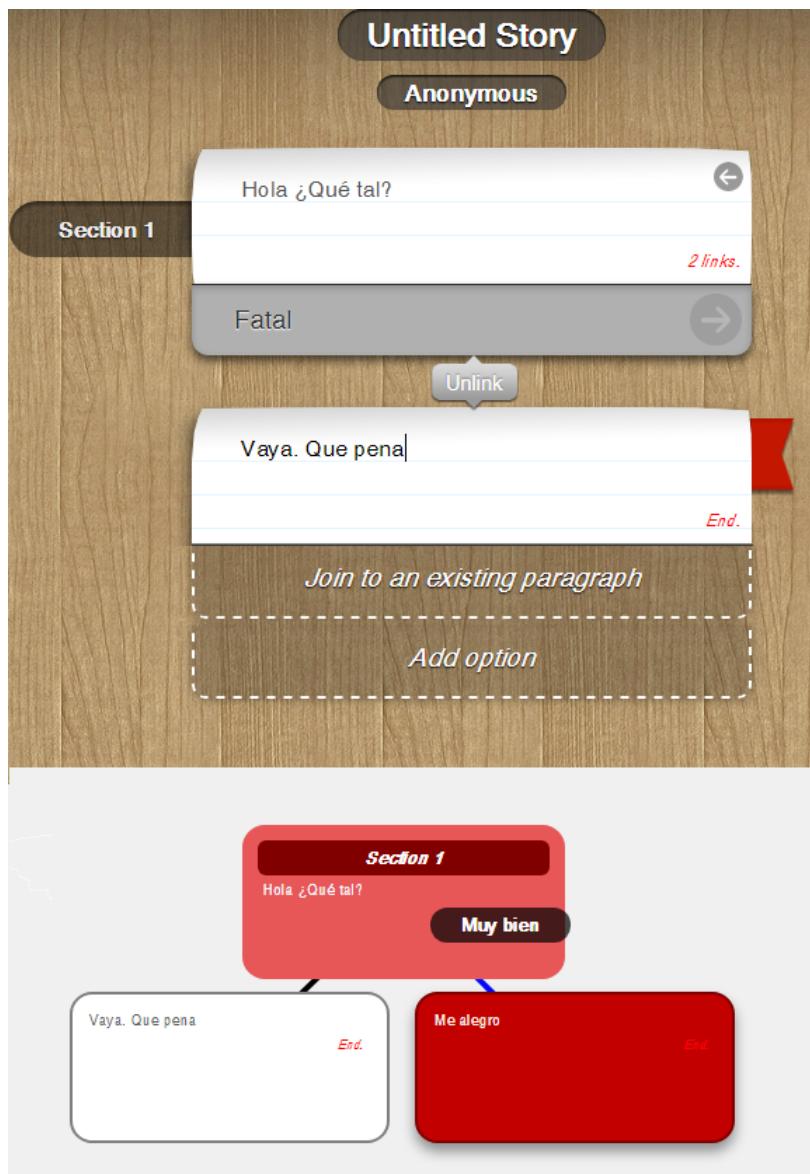


Figura 5.5.: Herramienta inklewriter

Para explicar la arquitectura del sistema de conversaciones se emplearán sendos diagramas de secuencia: uno para explicar el proceso de iniciar una conversación y otro para explicar el proceso de selección de contestaciones.

En el siguiente diagrama (ver fig. 5.6) se ve el flujo de código necesario para iniciar una conversación.

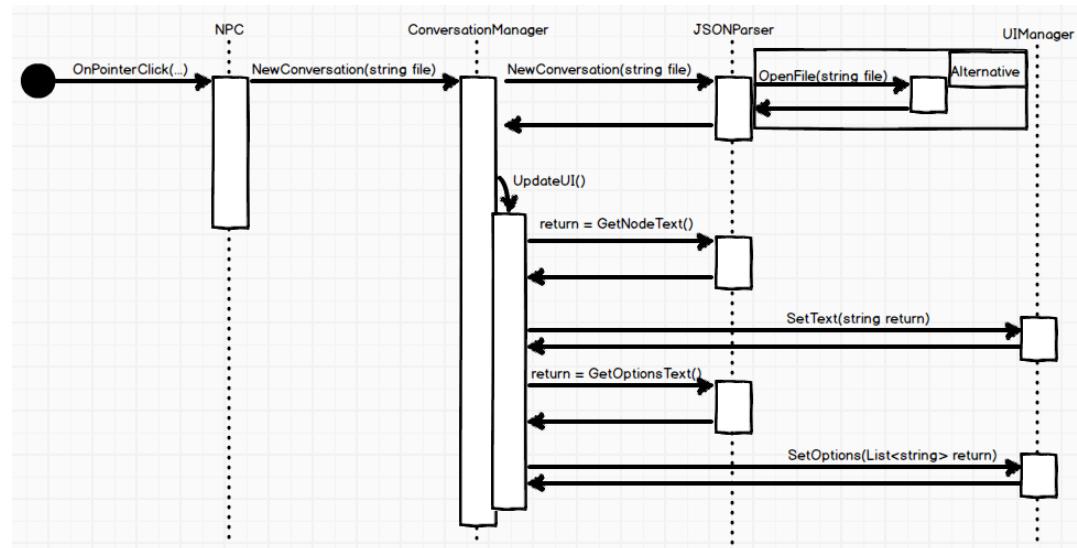


Figura 5.6.: Diagrama de flujo nueva conversación

No se ha indicado en el diagrama pero para el tratamiento JSON del fichero y para todas las operaciones JSON se utiliza la librería SimpleJSON¹⁰.

El punto de entrada es el click del jugador en un NPC. Previamente el NPC se habrá registrado en el ConversationManager, de forma que el NPC guarda en un delegado el método `NewConversation(string)` del ConversationManager.

Cuando se le pide al ConversationManager que inicie una nueva conversación, el NPC le pasa por parámetro la ruta del fichero en la que está almacenado su diálogo con el personaje que está controlando el jugador en ese momento. El ConversationManager informará al JSONParser de que se debe iniciar una nueva conversación, y este será el encargado de leer y parsear el fichero o de abrir una de las conversaciones que tiene almacenadas.

Una vez informado el JSONParser, el ConversationManager le pedirá el texto del nodo actual de la conversación y el texto de las posibles respuestas. Una vez tenga esa información se la pasará al UIManager que se encarga de la interacción con los elementos gráficos de la interfaz.

En este arquitectura se ha seguido un diseño inspirado en el patrón Model-view-Controller¹¹. Para ello todas las referencias a los elementos de la interfaz y los métodos de interacción con los mismos se han encapsulado en la clase llamada UIManager. El acceso a los datos de las conversaciones, la gestión del estado de las mismas y la lectura de los ficheros de diálogo se guardan en la clase JSONParser, que además utiliza la librería SimpleJSON. En cuanto al punto de encuentro entre los datos y la interfaz, se emplea

¹⁰<http://wiki.unity3d.com/index.php/SimpleJSON>

¹¹<https://es.wikipedia.org/wiki/Modelo-vista-controlador>

la clase ConversationManager para sincronizar las operaciones entre las otras dos clases y para proveer un punto de entrada al sistema de conversaciones.

Una vez abierta la conversación es necesario actualizar el juego cada vez que el jugador selecciona una respuesta. El funcionamiento de esta mecánica es muy similar y utiliza los mismos componentes que el sistema anterior. En el siguiente diagrama de flujo (ver fig. 5.7) se explica el proceso.

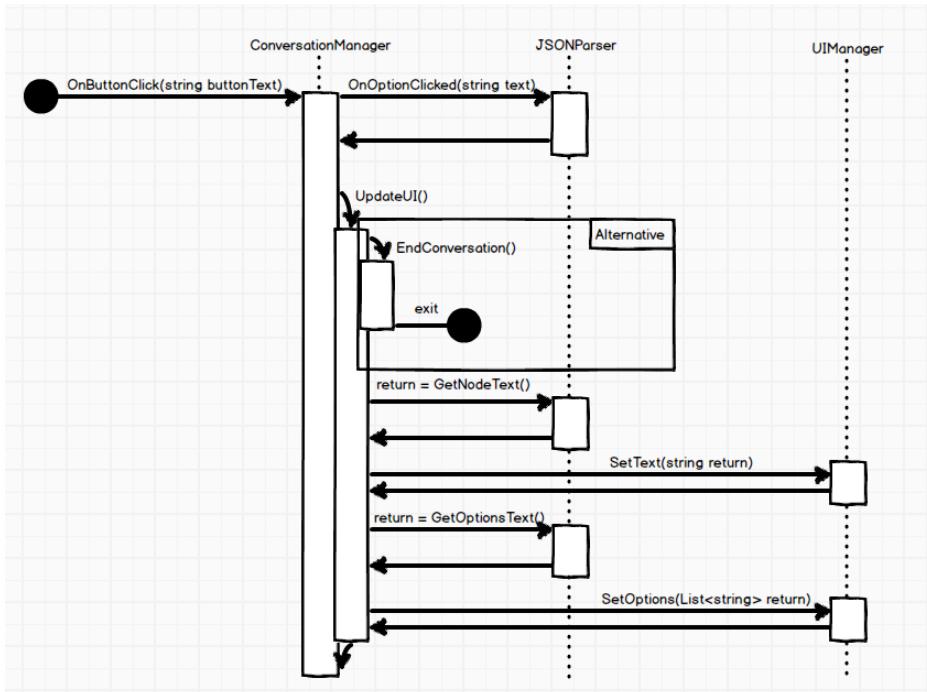


Figura 5.7.: Diagrama de flujo de selección de respuesta

Sistema de escenarios

A lo largo del juego se puede navegar por diferentes escenarios. Por temas de eficiencia y debido a que cada escenario tiene características únicas se han agrupado según la temática. En lugar de tener un gran escenario con montones de mallas correspondientes a los edificios y los personajes, solo están activos los Gameobjects correspondientes al escenario actual, los demás están desactivados (que no eliminados). En los escenarios hay puertas que al ser clicadas transportan al personaje activo hacia el escenario al que conduce la puerta.

En el inspector de escena de Unity3D dichos escenarios lucen como se en la siguiente imagen (ver fig. 5.8).

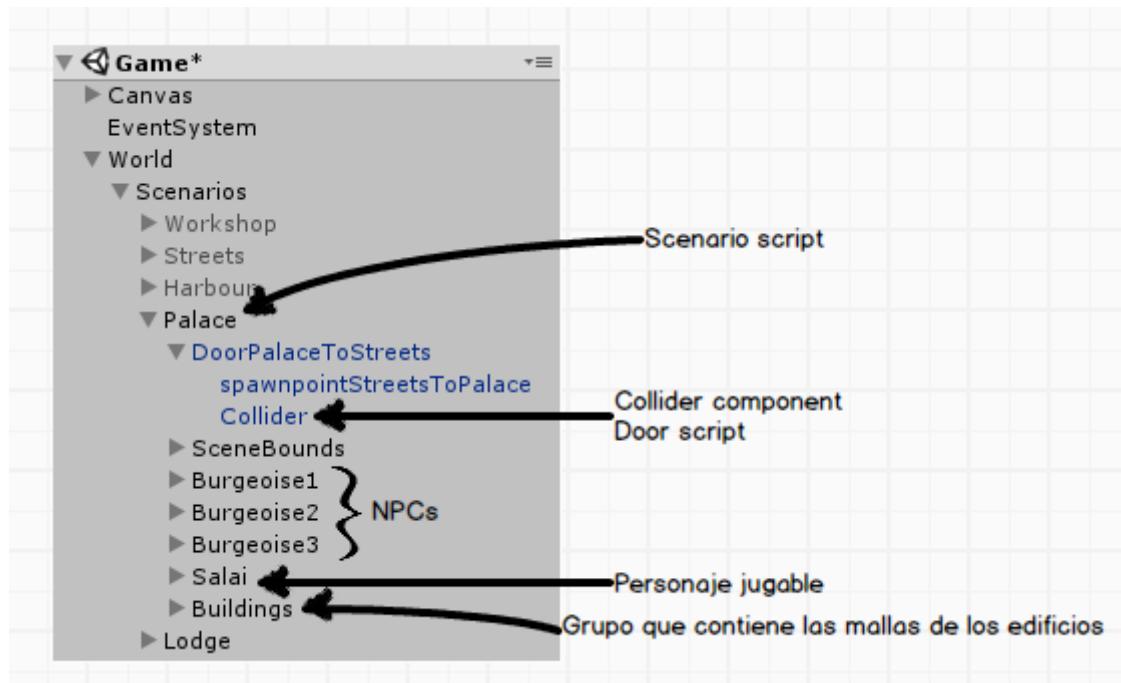


Figura 5.8.: Vista de los escenarios en el inspector de Unity3D

Como se puede observar algunos Gameobjects de la escena contienen scripts y componentes (otros Gameobjects tambien pero se han omitido para mayor simplicidad). En el siguiente diagrama de clases (ver fig. 5.9) se muestra la estructura de estos scripts y posteriormente se explicará su funcionamiento y utilidad.

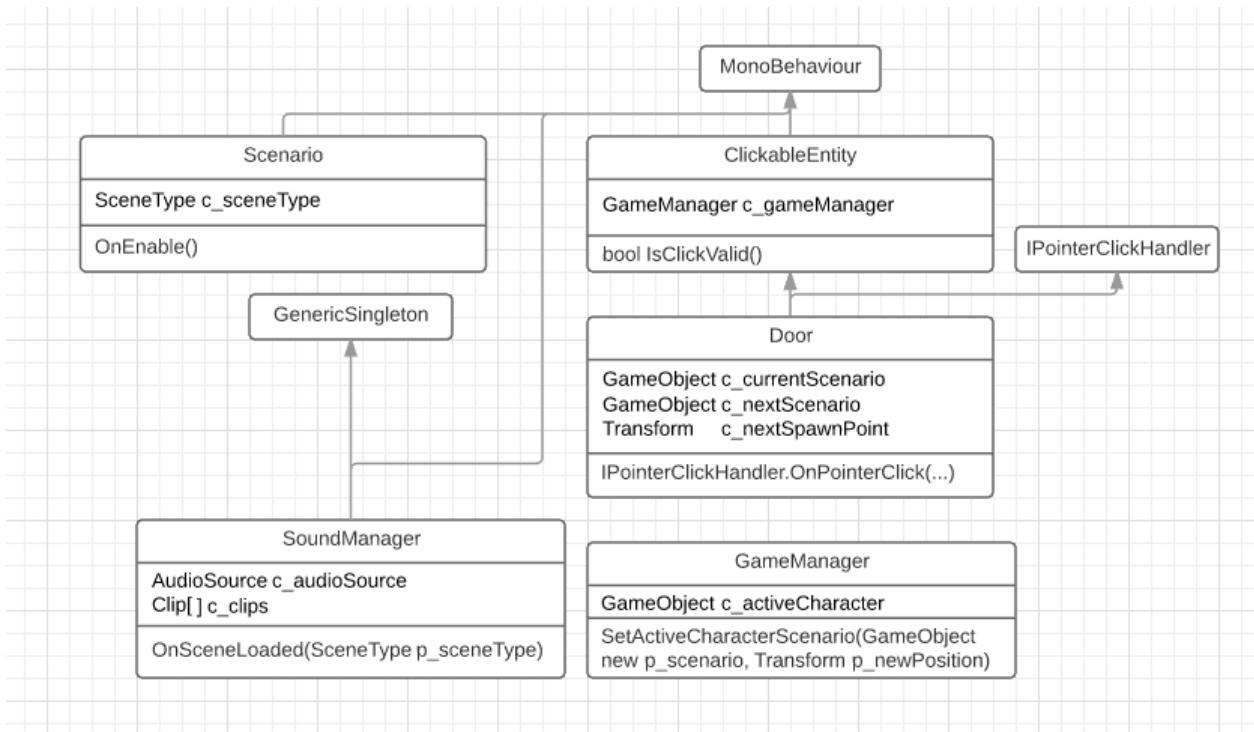


Figura 5.9.: Diagrama de clases del sistema de escenarios

El elemento central de todo el sistema de escenarios es la puerta. El nombre del Gameobject puerta sigue una notación que indica el escenario destino y origen. En el caso de la imagen anterior (ver fig. 5.8) se puede observar que la puerta lleva del Palacio (el escenario activo) a Las calles (un escenario desactivado).

El método `IPoINTERCLICKHANDLER.OnPoINTERCLICK(...)`, heredado de la interfaz `IPoINTERCLICKHANDLER` que provee Unity3D, es llamado cuando se hace click en el componente Collider asociado a la puerta. Posteriormente se comprobará que el click es válido (el jugador no está muy lejos por ejemplo) con el método heredado `IsClickValid()`. Finalmente se notificará al `GameManager` de que se quiere hacer un cambio de escena y se le suministrará el `GameObject` que contiene la nueva escena y la posición de inicio para el jugador.

El sistema de escenarios se encarga también de gestionar la pista de audio que se reproduce a modo de música ambiental. En el objeto raíz de cada escenario se encuentra un script que hace uso del método provisto por Unity3D, `OnEnable()`. Dicho método es llamado cada vez que se activa un escenario. Desde el método `OnEnable()` se notifica al `SoundManager`, que es accesible de forma global ya que implementa el patrón Singleton, el nuevo escenario activado. El `SoundManager` dispone de un clip de sonido por cada escenario y cuando recibe el tipo del nuevo escenario cambia el clip de sonido reproducido

por el componente AudioSource.

5.8.3. Storytelling

El storytelling es una parte fundamental del videojuego creado. Uno de los aspectos que hacen diferente a Da Vinci Startup es que en lugar de saturar al usuario con insufribles lecciones de teoría sobre emprendimiento, se aprende a la vez que se juega. Eso no significa que el conocimiento técnico no esté presente. La diferencia es que el conocimiento se adquiere a lo largo de la historia que los NPCs cuentan y que el jugador experimenta. De este modo conceptos como el "networking" no son solo palabras en un manual, si no que el jugador deberá utilizarlo a lo largo del juego para lograr los objetivos que le llevarán a completar el mismo.

En el Guion literario contenido en el Anexo I. Documento de diseño de videojuego se explica de forma narrativa la historia en la que se basa el videojuego. Es una historia ficticia en la que un mentor guía a Leonardo durante la construcción de la startup.

6. Resultados

Se puede afirmar que llegados a la finalización del desarrollo del sistema se ha obtenido un resultado más que satisfactorio. Se han implementado casi todos los requisitos del sistema y el videojuego provee de una experiencia de juego completa.

Se ha completado también el objetivo principal del proyecto: crear un videojuego que eduque sobre emprendimiento y «LeanStartup». Se ha usado el storytelling para que el jugador reciba información sobre «Lean startup» al conversar con los NPCs del juego (ver fig. 6.1).

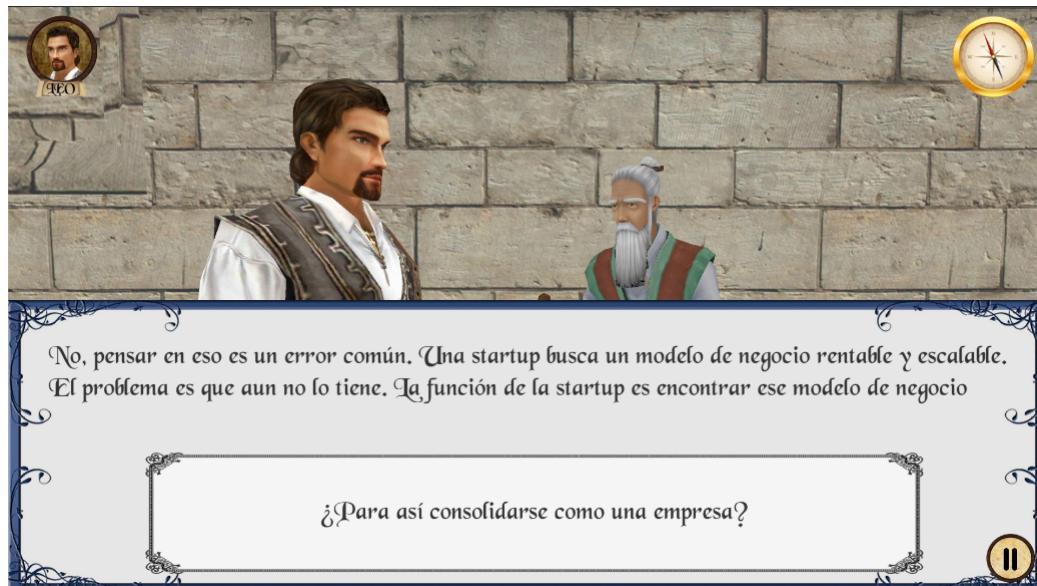


Figura 6.1.: Jugador dialogando con NPC en Da Vinci Startup

El desarrollo del proyecto se ha llevado a cabo con Unity3D y utilizando el lenguaje de programación C#. Se han ampliado enormemente los conocimientos sobre ambas herramientas y se han aprendido muchas técnicas de desarrollo sobre las mismas.

Se han creado también varios escenarios por los que el jugador puede transitar libremente y moverse de uno a otro (ver fig. 6.2).



Figura 6.2.: Jugador en el escenario Calles de Florencia

Se ha implementado también el sistema de interfaces propuesto, ofreciendo al jugador toda la funcionalidad necesaria pero sin comprometer la jugabilidad. Entre las interfaces creadas se encuentran el selector de personajes, el indicador de objetivos y el menu de pausa (ver fig. 6.3). Además se ha creado un menú principal con varias ventanas en las que visualizar información como los logros, las opciones, información sobre el desarrollador, etc (ver fig. 6.4).

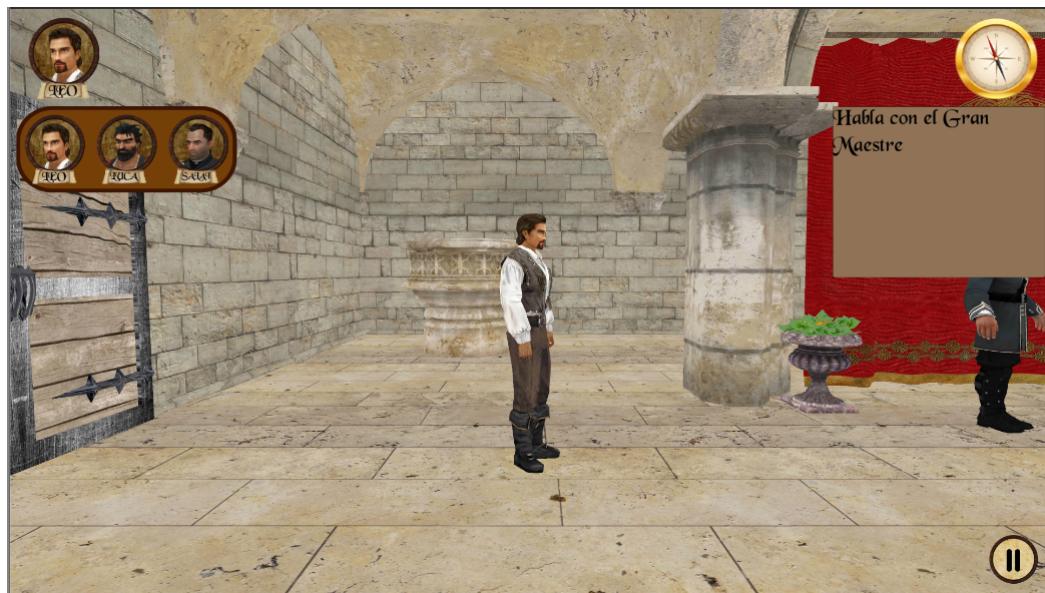


Figura 6.3.: Interfaz de usuario durante la partida con los menús desplegados

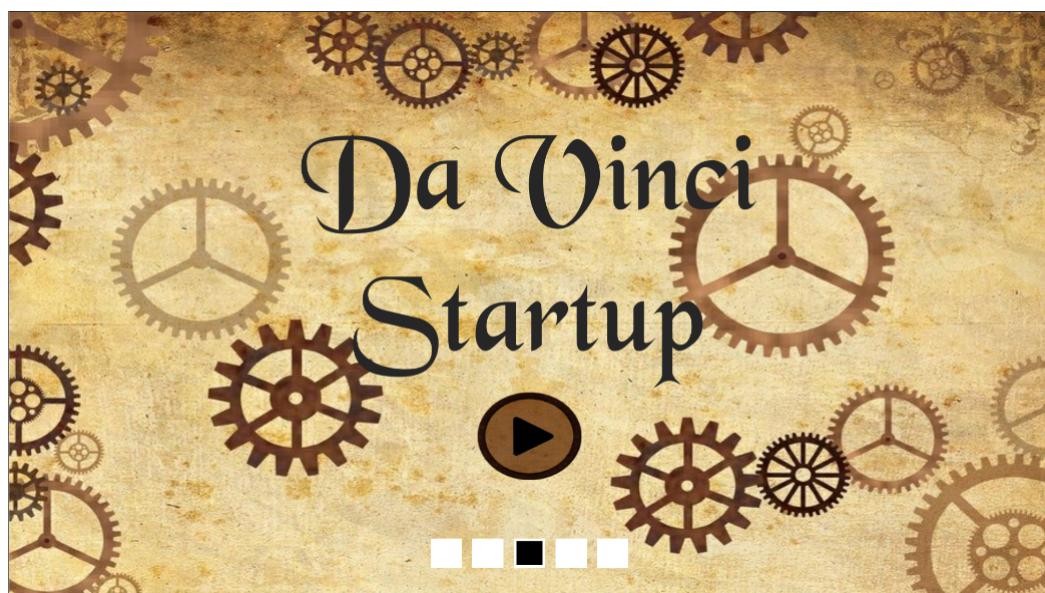


Figura 6.4.: Menú principal del juego

En cuanto a los objetivos que no se han podido cumplir: no se han podido crear los modelos 3D para el juego. En su lugar se han tenido que utilizar recursos que se ofrecen de forma gratuita. Tampoco se han podido cumplir los requisitos correspondientes a Guardar juego (RF-SYS-08) y Cargar juego (RF-SYS-09). En cualquier caso no son requisitos que comprometan la experiencia de juego y que impidan el transcurso de una partida.

En resumen, el producto creado no es un juego finalizado, pero si que se puede considerar un Mínimo producto viable que se podría enseñar a inversores que estén interesados en apoyar el desarrollo del juego. O mostrar a personas que podrían estar interesadas en unirse al equipo. También es el primer paso para poder probar el juego con usuarios reales y seguir trabajando acorde a las opiniones de estos.

Para concluir, el producto creado cumple perfectamente con lo esperado del mismo. Ofrece una experiencia de juego completa y contiene todos los elementos planeados para el juego. Es un punto de partida perfecto para reunir un equipo, buscar financiación y convertirlo en un juego que se pueda comercializar.

7. Pruebas y validación

Para el desarrollo no se ha optado por la escritura de tests automáticos ya que el tiempo requerido para este proceso habría sido inasumible. La utilización de tests automáticos es un proceso imprescindible en la ingeniería de software pero lamentablemente para este proyecto no ha sido posible.

Sin embargo sí que se han ejecutado diversas pruebas para comprobar que el sistema se comporta de forma adecuada y cumple con unos criterios de calidad.

7.1. Pruebas exploratorias

El la forma de testing más sencilla. Jugar el juego haciendo especial incapié en la búsqueda de bugs o errores. De esta forma se han encontrado multitud de comportamientos no deseados. Ha sido una de las formas de testing más utilizadas en este proyecto.

7.2. Pruebas funcionales

Haciendo uso de los requerimientos especificados en el apartado Requisitos de usuario se ha comprobado que estaban correctamente implementados mediante la interacción directa con la aplicación final. De esta forma se validan dichos requisitos ya que se prueban tal cual lo haría un jugador. Se han validado todos los requisitos implementados y se ha comprobado que funcionan de forma correcta.

7.3. Pruebas de usabilidad

Haciendo uso de un grupo de voluntarios que deseaban probar el juego se les dejó jugar durante un tiempo para luego realizarles una serie de preguntas. Se les pidió que valoraran la sencillez con la que se entendía las funcionalidades que provee la interfaz. Se les pidió también que valoraran la facilidad con la que el juego indicaba al jugador lo que debía hacer. La última batería de pruebas fue pedir a los usuarios que realizaran tareas sencillas tales como cambiar de personaje o cambiar de escenario. El aspecto a evaluar era el tiempo que tardaban en completar la tarea y cuanto les había costado descubrir como realizarla.

En general el resultado de las pruebas fue satisfactorio. Las interfaces tienen el mínimo de elementos posibles y las mecánicas de juego tampoco son muy elevadas por lo tanto es sencillo desenvolverse con el sistema.

7.4. Pruebas de compatibilidad

El sistema se probó en diferentes dispositivos con tamaños de pantalla y resoluciones variadas. El objetivo era verificar que las proporciones del juego, la estructura de la interfaz y de los escenarios no se veían afectadas por la resolución del dispositivo. Los dispositivos probados fueron un Xiaomi redmi3 con el sistema operativo Android y una resolución de pantalla de 1920x1080 en 5.5 pulgadas y una tableta Nexus 7 con el sistema operativo Android y una resolución de 1280x800 en 7 pulgadas.

En ambos dispositivos los resultados fueron los esperados: la interfaz se adapta a la pantalla perfectamente.

8. Conclusiones

8.1. Mejoras y ampliaciones

Como se ha indicado en el apartado anterior el producto creado no es un juego finalizado si no un Mínimo producto viable. Es por el ello que aún hay trabajo por delante hasta que se convierta en un producto comercializable

Las siguientes son algunas de las posibles mejoras al sistema:

- Implementar el sistema de guardado y cargado de partida.
- Incrementar la calidad/complejidad/cantidad de los diálogos.
- Utilizar modelos 3D propios.
- Aplicar animaciones a los modelos 3D

8.2. Modelo de negocio

Existen muchos métodos a la hora de comercializar un videojuego para plataformas móviles. En la actualidad las más usadas son la inclusión de publicidad, las compras dentro de la aplicación y la venta del videojuego en sí. Se ha decidido tomar la última opción por los siguientes motivos:

- La inclusión de publicidad distrae la atención del jugador de lo que realmente importa: el juego. Además dependiendo de lo intrusiva que sea esta puede ser incluso molesta y entorpecer la experiencia de juego.
- Las compras dentro de la aplicación no se pueden llevar a cabo ya que tal cual ha sido diseñado el juego no hay elementos que se puedan comprar, ni con dinero real ni con dinero del juego.

De este modo se empleará un enfoque comercial en el que se venderá el videojuego y el usuario no tendrá que lidiar con publicidad o con contenido extra que se desbloquee pagando. Tendrá todo el potencial del juego en una única compra.

Debido a que el juego puede ser una herramienta muy útil en cursos de formación emprendedora se ofertará la compra de paquetes de licencias. De esta forma los organizadores del curso podrán comprar paquetes con licencias, que evidentemente serán mas baratas que licencias individuales, y repartirlas entre los asistentes al curso.

También se pondrá a la venta en tiendas de aplicaciones como App store o Google Play para que cualquier persona, emprendedora o no, interesada en el juego pueda comprarlo.

En la siguiente imagen (ver fig. 8.1) se muestra un Lienzo de modelo de negocio con una propuesta para la comercialización del producto.

PROBLEM Para emprender es necesario formación académica. Los métodos de formación tradicionales resultan ineficaces para muchas personas. Los profesores sobre emprendimiento buscan nuevas alternativas para enseñar EXISTING ALTERNATIVES Ejercicios en grupo. Juegos de mesa	SOLUTION Ofrecer conocimiento a la vez que una experiencia de juego divertida. Un videojuego resulta más interesante que un libro de texto Es una forma de enseñanza muy novedosa	UNIQUE VALUE PROPOSITION Un videojuego para móviles que proporciona una experiencia de juego divertida y adictiva a la vez que enseña conceptos clave sobre emprendimiento y Lean Startup HIGH-LEVEL CONCEPT	UNFAIR ADVANTAGE Un videojuego que es realmente divertido y rompe con las formas tradicionales de enseñanza CHANNELS Tiendas de aplicaciones como Google play o App store.	CUSTOMER SEGMENTS Personas entre 20 y 50 años interesadas en emprender y en formarse sobre el tema. Organizadores de cursos de emprendimiento que buscan nuevas herramientas para enseñar EARLY ADOPTERS Interesados en la tecnología, los videojuegos y el emprendimiento. Personas que buscan formas alternativas a los métodos de enseñanza tradicionales
COST STRUCTURE Desarrollo del videojuego Publicidad			REVENUE STREAMS Venta de packs de licencias para poder jugar el juego. Venta de licencias individuales.	

Figura 8.1.: Lean canvas propuesto para este proyecto

Bibliografía

- [Albertini, 2015] Albertini (2015). Bienvenidos a catán, la isla que jubiló a un dentista.
- [antevenio, 2016] antevenio (2016). ¿en qué consiste la metodología lean startup?
- [Blank and Dorf, 2013] Blank, S. and Dorf, B. (2013). *El manual del emprendedor*. Grupo Planeta Spain.
- [del Bosque, 2016] del Bosque, D. (2016). 7 juegos virtuales que te enseñarán sobre negocios.
- [e mooc,] e mooc. Iterar o pivotar.
- [Entrepreneurship, 2016] Entrepreneurship, F. (2016). 7 juegos de mesa para emprendedores.
- [Fraga, 2016] Fraga, A. I. (2016). Así laten las startups en españa.
- [González, 2015] González, M. (2015). Si te gustó 'los colonos de catán', atento a estos dos nuevos juegos de mesa.
- [Martínez, 2014] Martínez, E. (2014). Las 8 grandes ventajas de las metodologías ágiles.
- [Navale, 2013] Navale, A. B. (2013). Developing entrepreneur skills for corporate work. *Research Directions*, 1:1–3.
- [Pastrana, 2015] Pastrana, O. (2015). 5 beneficios de aplicar metodologías ágiles en el desarrollo de software.
- [Peláez, 2015] Peláez, A. (2015). ¿en qué consiste la metodología lean startup?
- [Startupxplore, 2017] Startupxplore (2017). A view of the spanish startup community.

A. Anexo I. Documento de diseño de videojuego

Visión general

Introducción

Este juego recrea una aventura ficticia del famoso inventor Leonardo da Vinci.

A lo largo del juego, el inventor deberá desarrollar un producto, obtener financiación y venderlo siguiendo la metodología Lean Startup. Para ello contará con los consejos de Andrea del Verrocchio que será su mentor en el mundo del emprendimiento y las startups.

El juego será una aventura gráfica al estilo del mítico juego Monkey island, aunque se desarrollará en un escenario 2.5D como en el título Deadlight.

Tema

El juego se desarrolla en la Florencia renacentista, época en la que Leonardo trabajó en el taller de Andrea de Verrocchio. A lo largo de la partida se visitarán diversos escenarios como el taller de Andrea, las calles de Florencia o los palacios de los burgueses.

Estilo visual

Se utilizarán modelos 3D y texturas fotorealistas. No es necesario un gran nivel de detalle y/o de realismo pero se evitará una apariencia de estilo cartoon, comic o cel shading.

Influencias

Monkey island

Monkey island es uno de los referentes en cuanto a aventuras gráficas se refiere. Da Vinci startup está inspirado en este juego y utiliza gran parte de sus mecánicas. Por ejemplo los diálogos interactivos en formato árbol, en los que las decisiones tomadas por el jugador condicionan las respuestas de los personajes del juego.

Deadlight

Este juego posee un estilo similar al deseado en Da Vinci startup: un entorno 3D en el que el jugador se mueve en un plano 2D.

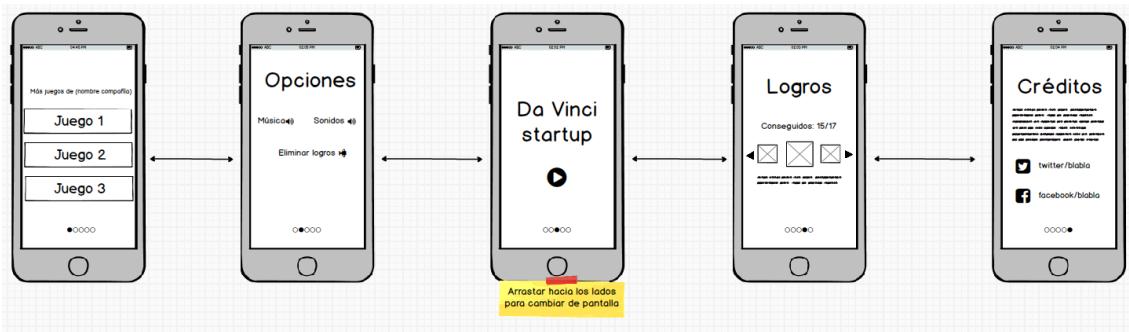


Menús

Menú principal

Es el menú que aparece inmediatamente después de que se abra el juego y el logo “Made with Unity” aparezca. En este menú se pueden acceder a todas las opciones disponibles del juego. La ventana que aparece inicialmente es la que contiene el título “Da Vinci startup” y para navegar hacia el resto se debe de arrastrar la pantalla hacia derecha o izquierda.

Si se clica en el botón play que hay en la pantalla inicial se empezará el juego.



Logros

Muestra los logros que han sido obtenidos durante el juego. En una línea se muestra cuántos logros se han conseguido y el total de los mismos.

Bajo esta línea se pueden ver las imágenes de los logros junto con un texto descriptivo. Los logros no completados mostrarán su imagen en blanco y negro, mientras que los sí completados la mostrarán a color.

A ambos lados de la hilera de imágenes se dispondrán de sendos botones con forma de flecha que al ser clicados cambiarán el logro que se está seleccionando en ese momento.

El logro seleccionado se muestra con un tamaño mayor. El texto descriptivo que aparece bajo la hilera de imágenes es el correspondiente al logro seleccionado.



Créditos

En un cuadro de texto se da crédito a las personas necesarias: autores de scripts, imágenes, música u otros elementos utilizados.

Se añaden también métodos de contacto con el autor.



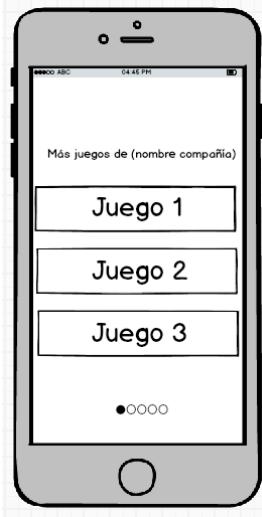
Opciones

En este menú se pueden cambiar diferentes aspectos del juegos tales como el volumen de la música o de los efectos sonoros o eliminar los logros obtenidos.



Otros juegos

Menú donde se pueden ver otros juegos creados por el autor. Al clicar en ellos el jugador es redirigido a la página correspondiente en Google Play.



Juego

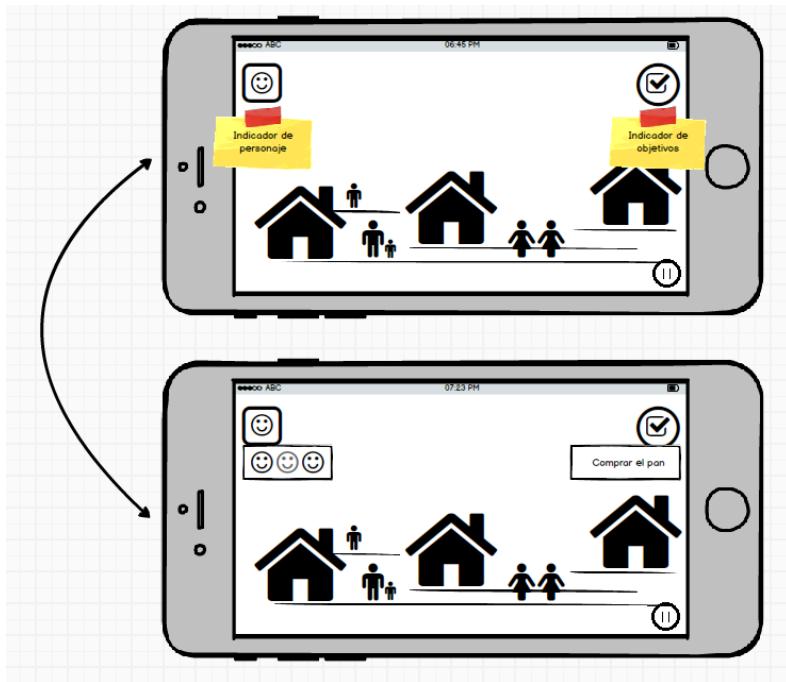
Durante el juego se dispone de una interfaz con 3 botones que son el Indicador de personaje, el Indicador de objetivos y el botón de pausa.

El indicador de personaje muestra una imagen del personaje (de los tres personajes controlables) que controla actualmente el jugador. Al clicar en el Indicador de personaje este se expande, y al hacerlo se muestran las caras de los tres personajes controlables. La cara del personaje que el jugador controla en ese momento se mostrará en gris.

Clicando en el icono de alguno de los otros jugadores hará que el menú se contraiga de nuevo y el personaje controlado cambie al seleccionado.

El Indicador de objetivos muestra el objetivo actual que se debe cumplir. Al clicar en el ícono del Indicador de objetivos se muestra una lista desplegable el texto descriptivo del objetivo.

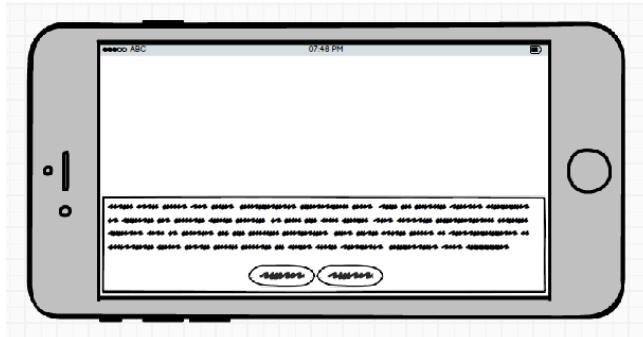
Al clicar en el botón de pausa se expandirá dicho menú mostrando las diferentes opciones que ofrece y se parará el juego.



Menú conversacional

Este menú se abre cuando el jugador interactúa con un personaje. La respuesta del personaje con el que se está interactuando se muestra en el texto grande, mientras que las posibles respuestas que puede dar el jugador se muestran en los botones.

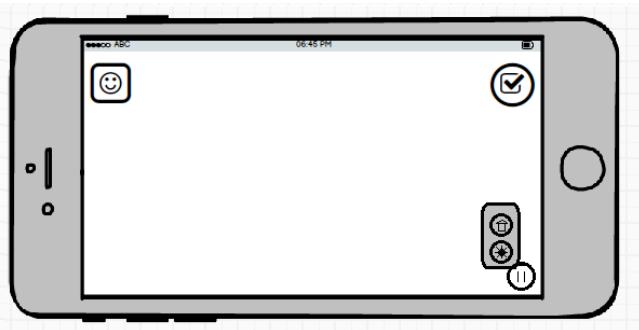
Cuando se da una respuesta con uno de los botones se actualiza la respuesta del personaje interactuado y el texto de los botones.



Pausa

Al clicar en el botón de pausa situado en la esquina inferior derecha el juego se pausará. También se abrirá un menú desplegable en el que el jugador podrá seleccionar entre volver al menú principal o abrir el menú de opciones.

Clicando nuevamente en el botón de pausa se cerrará el desplegable y el juego se reactivará.



Personajes

Controlables

Cada uno de los personajes controlables tiene una fortaleza y debilidad en forma de NPCs con los que pueden negociar. Durante el juego será necesario controlarlos a todos para poder negociar con los NPCs necesarios para avanzar en la historia.

Leonardo da Vinci

Protagonista de la historia. Se encarga de dialogar con Andrea del Verrocchio y es el ingeniero del taller. Es respetado en la Logia de ingenieros por lo que le escucharán cuando vaya. No puede entrar en el puerto y los burgueses no negocian con él.

Luca Pacioli

Tiene un aspecto desgarbado. Es uno de los ayudantes de Leonardo durante el juego. Puede entrar en el puerto y negociar con los capitanes y marineros pero no puede negociar con los burgueses ni con los ingenieros de la Logia.

Salai

Luce un aspecto elegante. Los burgueses negocian con él pero no puede entrar en el puerto ni negociar con los ingenieros de la Logia.

No controlables

Desarrollador

Ataviado con ropas y complementos del siglo XXI aparece en diversos lugares a lo largo del juego. Al hablar con él te pide que puntúes la app en Google Play, entre otras cosas.

Andrea del Verrocchio

Mentor de Leonardo. Da consejos sobre la metodología Lean startup. Se encuentra en el taller.

Marinero portero

Se encuentra en la entrada del puerto. Actúa como portero, dejando pasar solo a quien considera oportuno. Solo dejará pasar a Luca al puerto.

Marinero machaca 1

Puede ser contratado para trabajar en el taller de Leonardo. Se encuentra dentro del puerto.

Marinero machaca 2

Puede ser contratado para trabajar en el taller de Leonardo. Se encuentra dentro del puerto.

Capitán 1

No quiere comprar el invento pero está ansioso por venderte un barco. Está en el puerto

Capitán 2

No compra el invento pero te cuenta historias fantásticas sobre sus viajes en el mar. Está en el puerto

Capitán loco

Rechaza comprar el invento de Leonardo en primera instancia. Acepta comprarlo después de que pioche hacia algo que le interese más (barco propulsado por hélice). Se encuentra dentro del puerto.

Burgués 1

Acepta comprar el producto después de que lo hagan los capitanes, pero solo si se le añade una grúa para cargar mercancías. Está en el palacio.

Burgués 2

No le interesa el invento pero te vende bonos del estado a un precio bajísimo. Está en el palacio.

Burgués 3

No le interesa el invento pero te aconseja sobre apuestas deportivas. Está en el palacio.

Burgués 4

No le interesa el invento pero te cuenta historias sobre sus años de gloria cuando era rico. Está en el palacio.

Ingeniero 1

Es el Gran Maestre.

Ingeniero 1

Es un aprendiz de ingeniero. Se unirá el equipo de Leonardo si se le pide. Está en la logia.

Ingeniero 1

Es un ingeniero muy creído. Solo le interesa hablar de lo muy bueno que es y de los títulos que tiene. Está en la logia.

Ciudadano de Florencia 1, 2

Simplemente saluda.

Ciudadano de Florencia 3

Vende opio si le dices la contraseña correcta.

Ciudadano de Florencia 4

Pide dinero

Ciudadano de Florencia 5

Quiere que dones dinero para una asociación que suena muy falsa.

Mecánicas de juego

Movimiento

Los personajes controlables se pueden mover en el plano 2D. Para moverse, se toca en algún lugar de la pantalla y el personaje se moverá hacia la posición tocada (solo en el eje x).

Interacción con personajes

Al hacer tocar en la pantalla sobre un personaje el jugador se moverá hacia el personaje y al estar a su lado se abrirá el menú de conversación.

Seleccionar personaje

Durante el juego se pueden controlar tres diferentes personajes: Leonardo, Luca y Salai. En cualquier momento (siempre que no se esté en una conversación) se puede cambiar el personaje controlado usando el menú correspondiente. Para ello se deberá expandir el selector de personaje y clicar en el ícono de la cara del personaje deseado.

Conversar

Cuando se interactúa con un personaje se inicia una conversación. Para ello se abre automáticamente el menú conversación, que consiste en: se hace zoom sobre el juego hasta que los cuerpos de ambos personajes quedan en primer plano; se crea una ventana de texto en la parte inferior donde aparece el discurso del personaje que nos esté hablando; debajo del cuadro de texto aparecen botones con las posibles respuestas. Cuando se selecciona una respuesta se actualiza el cuadro de texto con una contestación y se actualizan también los botones de respuesta.

Mecánicas del mundo

Cambio de escenario

Cada escenario tiene puertas que lo conecta con los escenarios adyacentes. Por ejemplo, desde el Taller de Leonardo se puede acceder a las Calles de Florencia. A su vez desde las Calles de Florencia se puede acceder al puerto. El jugador solo puede estar en un escenario a la vez, y clicando en las puertas mencionadas anteriormente se cambia de un escenario a otro y se aparece en una posición concreta del escenario (un punto de spawn situado al lado de la puerta).

Scrolling paralax

Cuando el jugador se mueve por el mapa el fondo se desplaza también de forma horizontal. El fondo se divide en varias capas, cada una representando un plano a diferente distancia del jugador, y estas se desplazarán a diferente velocidad. Algunas de las capas se sitúan entre el jugador y la cámara, de forma que tapan la visión del jugador en ocasiones.

Completar objetivo

Tras realizar la acción asociada al objetivo actual, el ícono del indicador de objetivo parpadeará y variará ligeramente de tamaño de forma intermitente. Cuando el jugador abra el menú del indicador del objetivo aparecerá un nuevo objetivo y el anterior será eliminado.

Escenarios

Taller de Leonardo

En el taller se puede encontrar a Andrea del Verrocchio, que será quien dicte los objetivos a cumplir durante la historia y dará consejos sobre Lean startup.

Conecta con las Calles de Florencia.

Calles de Florencia

Las calles de Florencia es el escenario conector en el juego: desde las calles se puede acceder al resto de escenarios como el taller, el puerto, la logia o el palacio.

Los ciudadanos de Florencia pululan por las calles y el jugador podrá hablar con ellos.

Palacio de Lorenzo

En este edificio se encuentran los burgueses, que solo harán negocios con Salai.
Se accede desde las calles de Florencia.

Logia de ingenieros

En este edificio se encuentran los ingenieros, que solo hablarán con Leonardo. Se accede desde las calles de Florencia.

Puerto

Se encuentran marineros rudos que están ociosos. Un marino en la entrada del puerto actúa como portero dejando entrar a quien considera oportuno. Salai no puede entrar debido a que por su aspecto de clase alta los marineros no le dejan pasar. Leonardo no puede entrar ya que el puerto es un lugar peligroso y los marineros no le dejan pasar por si le ocurre algo.

Los marineros pueden ser contratados por Luca para trabajar en el taller.

Guion literario

Solo los ingenieros más sobresalientes son nombrados Gran maestre en la logia de ingenieros de Florencia. Esta es la ambición de Leonardo da Vinci, el más grande de los ingenieros de Florencia. Ha construido multitud de artefactos pero todavía tiene un último reto por delante antes de pasar de aprendiz a Gran maestre: debe construir un invento que se venda por un millón de florines.

Leonardo está contemplando el vuelo de un pájaro en las calles de Florencia y queda fascinado por la facilidad con la que la criatura se eleva hacia los cielos. Se le ocurre que si consiguiera construir un artefacto que pudiera hacer volar a las personas de la misma forma que lo hacen las aves se haría rico y se convertiría en un Gran Maestre. Llamará a este invento “el vuelacóptero”.

Con esta idea en mente va corriendo al taller en el que trabaja junto con su maestro Andrea del Verrocchio y sus aprendices Salai y Luca Paccioli. Una vez allí le cuenta la idea a Verrocchio y este le promete su ayuda a lo largo de todo el proceso de construir y comercializar el invento puesto que es un gurú de una nueva metodología de trabajo inventada en “Toscana Valley”.

Leonardo está ansioso por ponerse manos a la obra pero Verrocchio le advierte de que para que un proyecto funcione no se puede trabajar a lo cowboy, hace falta un equipo que aporte los conocimientos que uno no tiene. Es necesario conocer gente, y eso solo se puede hacer saliendo a la calle. En Toscana Valley llaman a esto “hacer networking”. De esta forma la primera tarea del taller es encontrar a unos obreros que ayuden a construir. Estos obreros serán marineros parados del puerto, al que solo es permitido el acceso a Luca.

Una vez conseguida la mano de obra es necesario conseguir financiación, para lo cual se pueden tomar diferentes alternativas que Verrocchio explicará: se puede optar por el bootstrapping o buscar financiación externa. El taller deberá optar por la segunda opción ya que con el capital que tienen no pueden asumir los costes del proyecto. Para conseguir financiación el lugar más adecuado es el palacio donde los burgueses se reúnen a discutir sobre economía. Solo Salai, con su estatus de burgués, conseguirá que los burgueses le tomen en serio y le proporcionen los fondos que necesitan.

Conseguidos financiación y mano de obra, solo falta hacer ingeniería y diseñar el producto que se va a construir. Para ello Leonardo deberá convencer a algún ingeniero de la logia para que se una al equipo, y una vez conseguido, trabajar en los planos.

Leonardo está impaciente, con todos los recursos reunidos solo falta ponerse manos a la obra y construir el vuelacóptero. Una vez construido todo el mundo querrá uno y se harán ricos. Craso error. Verrocchio le quita la idea rápidamente de la cabeza: así no se hacen las cosas en Toscana Valley. Las ideas suenan perfectas en nuestra cabeza pero rara vez coinciden con las de los clientes. Así que en lugar de construir el producto y esperar a que esté finalizado para venderlo, lo que harán será utilizar un desarrollo iterativo: construirán una versión temprana del vuelacóptero y le irán añadiendo partes a medida que los clientes digan lo que les gusta y lo que no.

Pasan unas semanas, la primera iteración del desarrollo ha sido completada y Leonardo va a hablar con Verrocchio para preguntar cuáles son los siguientes pasos. Ahora que tienen un MVP (producto mínimo viable) es hora de salir a la calle y conseguir lo que en Toscana Valley se llama “early adopters”: gente tan loca como ellos que se atrevan a comprar un producto novedoso aún sin finalizar.

La búsqueda es infructuosa y al parecer nadie quiere comprar un vuelacóptero, pero un capitán loco del puerto sugiere que con ciertos cambios tal vez estaría interesado.

Leonardo le cuenta sobre el desastre a Verrocchio y este le explica que gracias a que no han estado durante meses gastando dinero y construyendo una versión definitiva del producto, ahora pueden reaccionar a tiempo y cambiar lo que sea necesario para poder venderlo.

Cuando hay que hacer grandes cambios sobre la hipótesis de negocio, se habla de pivotar. De forma que realizando estos cambios la hipótesis de negocio evoluciona de acuerdo a la experiencia adquirida y se acerca más a un modelo de negocio viable.

De acuerdo a lo requerido por el capitán loco, incorporarán el vuelacóptero a un barco para que navegue más deprisa. Una vez construido el nuevo barco se le comunica al capitán loco y este lo compra muy gustosamente.

Leonardo le da la noticia a Verrocchio y ambos lo celebran, pero tienen que seguir consiguiendo ventas así que hay que volver a la calle. Al hablar con los burgueses a estos les parece interesante el producto pero no se acomoda del todo a sus necesidades: lo comprarán solo si dispone de una grúa para cargar y descargar mercancías.

Sabiendo esto, Verrocchio explica que el producto se tiene que adaptar a las necesidades y deseos de los clientes. Esto se denomina “customer development”. Los pequeños cambios en el modelo de negocio, como añadir la grúa al barco, se denominan pivotar.

Con los nuevos cambios en el barco los burgueses lo compran encantados y Leonardo consigue la meta que tenía que cumplir para convertirse en Gran maestre. Reflexionando con Verrocchio se da cuenta de cómo ha evolucionado la idea que tuvo originalmente (el vuelacóptero) hasta convertirse en el barco mercante propulsado por hélice. También piensan en cómo habría resultado todo si no hubieran empleado una metodología de desarrollo ágil y hubieran empleado muchos esfuerzos en construir el vuelacóptero antes de mostrarlo a los potenciales clientes.

Finalmente Leonardo sale del taller para ir a la logia de ingenieros y la puerta del taller está abarrotada de gente que quiere comprar uno de sus barcos. La fama de sus veloces barcos mercantes se ha extendido por toda Florencia. En la logia es convertido en Gran maestre y el ingeniero jefe le explica que la búsqueda de su startup ha finalizado. Han conseguido encontrar su hueco en el mercado y desarrollar su producto. Da Vinci startup deja de ser una startup.

Objetivos

El flujo del juego es controlado por el objetivo a cumplir. En todo momento se dispone de un objetivo a cumplir, y al hacerlo se desbloquea el siguiente y el mundo del juego se actualiza.

De este modo el transcurso del juego se basa en el cumplimiento de los objetivos y son el hilo conductor de la historia. La lista de los objetivos ordenados por orden de aparición es:

1. Habla con el Gran Maestre
2. Ve al taller y pide ayuda al maestro Verrocchio
3. Ve al puerto y consigue constructores

4. Ve al taller y habla con el maestro Verrocchio
5. Consigue financiación de los burgueses
6. Ve al taller y habla con el maestro Verrocchio
7. Consigue un ingeniero en la logia
8. Ve al taller y habla con el maestro Verrocchio
9. Busca early adopters
10. Ve al taller y cuentale el fracaso al maestro Verrocchio
11. Habla con el capitán loco
12. Ve al taller y habla con el maestro Verrocchio
13. Busca más early adopters
14. Ve al taller y habla con el maestro Verrocchio
15. Habla con el burgués
16. Ve al taller y habla con el maestro Verrocchio
17. Habla con el gran maestre

Logros

Además de los objetivos se podrán desbloquear logros, aunque el desbloqueo de estos no es requerido para completar el juego. Su función es meramente colecciónista. Los logros que se podrán desbloquear son:

- La ira de los bits caerá sobre ti
- Opio conseguido. Es hora de montar una buena fiesta
- Has conquistado el corazón de un developer
- Has completado el primer objetivo
- Has completado el juego

Assets requeridos

Modelos 3D

Personajes

Se necesitará un modelo por cada uno de los personajes del juego. Algunos de ellos reutilizarán el mismo modelo y solo cambiarán las texturas que utilizan.

Los modelos requeridos son:

- Leonardo
- Luca
- Salai

- Verrocchio
- Capitán (compartido por los dos capitanes y el capitán loco)
- Marinero (compartido por los dos marineros)
- Burgués (compartido por todos los burgueses)
- Ingeniero (compartido por todos los ingenieros)
- Desarrollador

Edificios

Se requerirán edificios para todos los escenarios del juego. En algunos de los escenarios (taller, puerto) se necesitará utilería que represente el trabajo que ahí se hace.

Música

Se requerirá una pista de sonido ambiente por cada uno de los escenarios:

- Taller de Leonardo
- Logia de ingenieros
- Calles de Florencia
- Palacio de Lorenzo
- Puerto

Además se incluirá una pista de sonido para el menú principal.

Sprites

IU

Durante el juego:

- Icono selector personajes
 - Icono cara Salai
 - Icono cara Luca
 - Icono cara Leonardo
- Icono indicador de objetivos
- Icono pausa
 - Icono home
 - Icono menú opciones

En el menú principal:

- Fondo de los diferentes menús
- Botón iniciar juego
- Iconos de logros

- Flechas de navegación entre logros
- Marco para texto de logros
- Icono Linkedin
- Iconos altavoces con diferentes volúmenes