Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de un videojuego multijugador local con Unity3D: implementación de la Inteligencia Artificial

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Marcos Crispín Ortuzar

Tutor: Patricio Letelier Torres

2016 - 2017

Resumen

El videojuego se ha convertido, durante los últimos años, en la industria del entretenimiento más potente y con más proyección de futuro. Cada vez son más los pequeños estudios que desarrollan sus ideas y se abren hueco en una industria de tal envergadura a base de pasión, creatividad y esfuerzo. Son esas características las que nos motivaron a dar forma y viabilidad económica a nuestras ideas.

Este TFG aborda el proceso de desarrollo de un videojuego multijugador desde la perspectiva de un proyecto de emprendimiento, abarcando desde el nacimiento de la idea base hasta la implementación del Mínimo Producto Viable, pasando por las fases de estudio de mercado y de pruebas de los prototipos.

Palabras clave: desarrollo de un videojuego, emprendimiento.

Abstract

The videogame industry has become in recent year the biggest and more promising entertainment industry. More and more small studios are starting to develop their videogame ideas, getting in on an industry of such magnitude through passion, creativity and hard work. Those are the elements that pushed us to craft and to give economic viability to our own ideas.

This paper is about the development process of one of those ideas in the form of a multiplayer videogame through the entrepreneurship project perspective, from the birth of the idea to the implementation of the minimum viable product, passing through the market analysis and the prototypes testing phases.

Keywords : videogame development, entrepreneurship.

A Zaida Abad, el mejor de mis recuerdos

Tabla de contenidos

1. Introducción

Figura 1: Logo del videojuego

Nos encontramos, sin duda, en la década donde los videojuegos han dejado de ser un entretenimiento digital secundario para alzarse como una de las industrias con mayor impacto económico actual. Sólo en Estados Unidos, la industria del videojuego ha pasado de generar 2 mil millones de dólares en 1998 y 6.9 mil millones en ventas en el 2005 a 17.1 mil millones en el 2010.[1]

Un factor determinante ha sido el ascenso de las plataformas de venta digital, tales como *Steam*[1], *Xbox Live* [2]o *Playstation Network*[3]. Estas plataformas han permitido que cada vez más estudios pequeños, con bajos presupuestos y personal, sean capaces de publicar y llegar al mismo número de usuarios que las grandes superproducciones, pudiendo incluso competir con ellas en cuanto a reconocimiento y ventas. Estos juegos de estudios menores son conocidos como videojuegos independientes (*indie* como término coloquial).

Este trabajo final de grado aborda el proceso de desarrollo de un videojuego independiente abarcando desde su idea inicial hasta la puesta en marcha de una campaña de financiación por *crowdfunding*[4]. Para lograr dicho objetivo, se ha dividido el proceso en cuatro etapas:

- Estudio de mercado y definición de la proyección económica del proyecto
- Desarrollo de un MVP (mínimo producto viable)
- Pruebas con *early-adopters*[5]
- Publicación de la campaña de financiación.

[1] http://store.steampowered.com
[2] http://www.xbox.com/en-US/live
[3] https://www.playstation.com/es-es/explore/playstation-network/
[4] Modelo de financiación colectiva a través de donaciones
[5] Consumidores del producto antes de que se encuentre en el mercado

El videojuego a desarrollar tiene como título "*Gnomore Gnomes*", y está desarrollado sobre el motor gráfico *Unity 3D*. La plataforma inicial para su publicación es *Steam*, presente en *Microsoft Windows, Mac OSX y Linux* (en el capítulo dos de esta memoria se profundiza más en la plataforma). Sin embargo, en función del buen rendimiento de la campaña de financiación, no se descarta la posterior adaptación a videoconsolas de sobremesa, como *Switch,* de *Nintendo,* o *Xbox One*, de *Microsoft*.

Este proyecto ha sido desarrollado junto a Jesús Lor en el espacio emprendedor *Start.Inf* de la *Escuela Técnica Superior de Ingeniería Informática*. En esta memoria se tratará más específicamente el proceso de implementación de la inteligencia artifical de uno de los personajes del videojuego. Por otra parte, en el TFG de Jesús Lor se detallarán los aspectos asociados al diseño y animación de personajes y escenarios.

1.1 Motivación

La principal motivación para el desarrollo de este proyecto nace del deseo de tener la posibilidad de crear nuestro propio videojuego. Deseo presente desde nuestra niñez, cuando jugábamos por primera vez a los juegos que existían por aquella época. Fue ese deseo el que nos empujó a iniciar el grado en ingeniería informática, ya que sentíamos que podíamos aunar nuestro amor por la tecnología y dar rienda suelta a nuestra creatividad para crear mundos, personajes y contagiar ese amor por los videojuegos a todo aquel que tuviera un mando en sus manos y nuestro videojuego en su pantalla.

A día de hoy aún no podemos pensar en una mejor forma de ganarse la vida.

A pesar de ser es un tipo de entretenimiento relativamente joven (los primeros videojuegos datan de principios de los años 60), los videojuegos han sufrido una evolución constante que les ha permitido diversificarse en una gran variedad de géneros. Podemos encontrar desde simuladores militares o deportivos, a aventuras de rol u orientados al juego en familia.

Por ello, las opciones eran casi ilimitadas cuando nos enfrentamos a la decisión de qué videojuego queríamos hacer. Repasando nuestras propias experiencias como usuarios habituales de videojuegos, descubrimos que a pesar del paso de los años, permanecía la afición por los comúnmente llamados "videojuegos de sofá". Estos son aquellos juegos con un enfoque multijugador competitivo, en el que todos sus jugadores están conectados al mismo sistema de juego, ya sea un ordenador personal o una videoconsola. Por tanto, la experiencia de juego se reparte entre la que aporta el propio software, como la que se genera entre los jugadores de forma externa.

Los recuerdos de nuestras experiencias entre amigos y familiares, de las rivalidades, las burlas, la emoción de ganar y la frustración de perder una y otra vez frente al mismo amigo fue lo que nos impulsó a decantarnos por este estilo de videojuego, y así poder generar esas experiencias en todos los futuros jugadores de "*Gnomore Gnomes*".

1.2 Objetivos

El objetivo de este TFG es desarrollar una primera versión del juego *Gnomore Gnomes*. La idea es que este MVP (Mínimo Producto Viable) nos permita publicar el videojuego en la plataforma digital *Steam* en modo *Early-Access*. Este modo de venta permite a los usuarios comprar el juego a un precio reducido, sabiendo el comprador que el software tiene partes aún por completar. Esto permite al desarrollador obtener ingresos antes de que el producto esté completamente terminado. Para lograrlo, se han marcado los siguientes objetivos:

- Desarrollar un prototipo que permita jugar una partida completa con soporte para cuatro jugadores.
- Presentar el prototipo en algún evento de prueba de videojuegos en desarrollo.
- Publicar la campaña de financiación colectiva en la plataforma *Kickstarter*, que permita cubrir los costes de producción hasta que el videojuego esté a la venta en la plataforma de venta digital *Steam*.

1.3 Estructura de la memoria

La memoria se ha estructurado de la siguiente forma:

El primer capítulo introduce el tema del trabajo fin de grado, la motivación detrás de la elección del proyecto y los objetivos a cumplir al finalizarlo.

El segundo capítulo define el concepto general del videojuego, desarrollando la evolución que sufrió desde la primera idea original hasta el concepto final sobre el que está implementado el mínimo producto viable.

El tercer capítulo comprende el análisis del videojuego desde el punto de vista de mercado. Comienza introduciendo el estado del arte, haciendo un repaso de la historia de los videojuegos de corte multijugador local con especial hincapié en algunos de los referentes actuales del género, para terminar con una tabla comparativa con el presentado en este trabajo.

A continuación se realiza la proyección económica del proyecto, estableciendo las estimaciones de los costes y de los ingresos, exponiendo el *Lean Canvas* y el análisis *DAFO* (Debilidades, Fortalezas, Amenazas y Oportunidades) referentes al proyecto.

En el cuarto capítulo se desarrolla el proceso de desarrollo del videojuego, mostrando el mapa de características del videojuego, y cómo se fueron desarrollando cronológicamente, mostrando cómo afectaron las distintas pruebas con los *early-adopters* al mapa de características y a los plazos originalmente establecidos.

El quinto capítulo analiza las tecnologías utilizadas para el desarrollo del videojuego.

En el sexto capítulo se expone cómo está enfocado el desarrollo de la inteligencia artificial de uno de los personajes del videojuego, y los sistemas que intervienen en su implementación.

El último capítulo presenta las conclusiones, así como los planes de futuro para el videojuego.

2. Concepto del juego

El videojuego que se ha desarrollado a lo largo del proyecto se titula 'Gnomore Gnomes', (que surge de un juego de palabras entre los términos en lengua inglesa "Gnome" y "No more") que pertenece al género arcade [6] de carácter multijugador local para un máximo de 4 jugadores.

Como se indica en el apartado de motivación incluido en el capítulo de introducción, la idea alrededor del videojuego nace de la voluntad de desarrollar un videojuego donde prime la diversión y la competitividad entre jugadores que se encuentren en el mismo espacio físico. Uno de los factores que se tuvo en cuenta para establecer el concepto básico del juego fue el de minimizar todo lo posible la curva de aprendizaje del usuario. Esta característica es fundamental para juegos de esta índole, pues el juego está enfocado para partidas cortas entre varios jugadores. Es más que probable que cuando el usuario propietario del videojuego realice una sesión de juego, algunos de los restantes jugadores sean nuevos y no conozcan las reglas de este, o no estén familiarizados con el sistema. Siendo la base del software las partidas competitivas cortas, si miembros del equipo no saben qué hacer, ni cómo reaccionar a lo que están viendo por pantalla, la experiencia se empobrece y resulta frustrante para todos los usuarios que se encuentren jugando (tanto nuevos como habituales).

Por tanto se buscó definir unas mecánicas de juego lo más fáciles de comprender y asimilar, así como un uso del mando de juego sencillo e intuitivo (sin por ello perder identidad o características que hagan al juego divertido y atractivo para el jugador).

2.1 Idea inicial

La idea inicial se desarrolló en base a dos pilares iniciales:

- La temática del videojuego engloba la celebración de un torneo con ambientación de fantasía medieval, donde el objetivo fuera eliminar una serie de gnomos debido a una plaga que asola el ficticio reino.

- La mecánica jugable básica se compone de una serie de cortas y variadas pruebas a lo largo del torneo, estando diseñadas todas ellas con un claro enfoque hacia la acción para varios jugadores.

Después de varias sesiones de brainstorming[7], definimos la idea base sobre la que empezar a trabajar en profundidad las mecánicas jugables. Durante dichas sesiones, para diseñar la primera de las pruebas, se buscó reinterpretar la jugabilidad del deporte conocido como rugby, añadiendo y eliminando reglas, en favor de una jugabilidad simple y rápida.

[6] Género de videojuegos que sigue los principios de diseño de los antiguos videojuegos presentes en máquinas recreativas

[7] Técnica de grupo para generar ideas en un ambiente relajado

El concepto general giraba en torno a cuatro jugadores (sin subdivisión por equipos), que debían coger un gnomo que corría suelto por el terreno de juego. Una vez uno de los jugadores tenía el control de dicho gnomo, debía mantenerlo el máximo tiempo sin que otro jugador se lo arrebatara mediante un derribo. Para obtener puntuación, el personaje portador del gnomo debía ir golpeando al mismo, para así acumular puntos. Al finalizar el tiempo reglamentario, el personaje con mayor puntuación resultaba elegido ganador.

La idea sonaba prometedora, pues cumplía los requisitos que nos impusimos al comenzar el diseño de la jugabilidad: eran unas reglas sencillas de entender, con un número de acciones posibles para el jugador mínimo, que además mantenía el estilo de jugabilidad caótica y desenfrenada que buscábamos desde el principio. Decidimos utilizarlo como punto de partida y comenzamos el desarrollo del prototipo.

2.2 Evolución hasta la idea final

Como se explicará con detenimiento a lo largo del cuarto capítulo de esta memoria, durante el comienzo del desarrollo de la primera versión del prototipo, surgieron dos importantes cuestiones acerca del videojuego necesarias de resolver antes de continuar el desarrollo:

La primera tenía relación con los recursos y el tiempo disponible, pues comenzamos a observar que probablemente no sería posible llegar al final del desarrollo del Mínimo Producto Viable (alrededor de junio del año 2017) teniendo desarrolladas un número suficiente de pruebas del torneo como para aportar suficiente variedad y rejugabilidad al título. Las opciones a elegir eran escasas. La primera y más obvia era la adquisición de más recursos. Lo mínimo para llegar al plazo con el doble de pruebas era la inclusión en el equipo de un programador y un artista más. Dado que no teníamos un horario fijo de trabajo, ya que teníamos que estar acomodando los horarios de desarrollo a los horarios de las clases de la facultad o a trabajos fuera de ella, y que no disponíamos de financiación para cubrir los costes de dos participantes más, decidimos seguir buscando otra solución.

La segunda gran cuestión estuvo muy presente a lo largo de todo el proceso de desarrollo. Durante la partida, la función del jugador que tenía el control del gnomo tenía una función clara y desafiante, pues tenía que reaccionar en base de las acciones del resto de tres jugadores, para evitar perder la posesión del gnomo que porta. Sin embargo, la función del jugador que no tiene la posesión del gnomo, era excesivamente simple, pues no tenía más que perseguir al jugador que lo portaba. Como resultado, la acción de la partida se concentraba siempre en el punto en el cual se hallara el jugador portador, estando los tres jugadores restantes siempre pegados a él. Esto provocaba una situación frustrante continua, debido a que en cuanto un nuevo jugador obtenía el gnomo, pocos segundos después lo perdía, pues los otros tres

personajes ya se encontraban muy cerca de él para derribarlo. Mientras tanto, el resto del terreno de juego permanecía vacío.
Por tanto, era importante encontrar la manera de repartir a los jugadores a lo largo del terreno de juego, y, sobre todo, de hacer sentir a los jugadores que tenían un rol diferenciado sobre la partida (como pueden ser los roles de defensa, mediocentro y delantero en un partido de fútbol tradicional), y que necesitaban plantear estrategias y tomar decisiones de equipo para obtener la victoria.

Teniendo que encontrar una solución a ambos problemas, llegamos a la conclusión de que teníamos que replantear la base jugable de las pruebas si queríamos seguir siendo dos desarrolladores y mantener el grado de calidad del producto. La solución a la que se llegó fue otorgarle más profundidad a la prueba que nos encontrábamos diseñando, aportando más reglas y acciones para los jugadores y así pasar de un sistema de varias pruebas a un partido completo como si de un deporte común se tratara.

Esta decisión suponía la pérdida de la variedad inicial, pues sólo podía jugarse un tipo de partido. Para compensar la limitación de las mecánicas básicas, se optó por ampliarla presentando una serie de escenarios donde disputar los partidos, donde no sólo la estética fuera radicalmente distinta, sino que, además, cada uno de ellos aportara mecánicas jugables únicas a la base común, haciendo que los jugadores tuvieran que variar su forma de enfrentarse al juego dependiendo de cuál fuera el escenario donde se está dando lugar la partida.

Sólo faltaba aportar profundidad a las bases jugables del ‘deporte’ para que se sostuviera como pilar principal del juego. Durante el desarrollo del proyecto, estas fueron variando, evolucionando o siendo descartadas (Este proceso se detalla durante el capítulo cuatro de esta memoria). A continuación, se enumeran los cambios que se produjeron sobre la idea inicial, en su estado definitivo, como se muestra en la figura 2:

- Los jugadores pasan a estar repartidos en dos equipos de dos jugadores cada uno.

- La raza del personaje afecta a la jugabilidad, aportando más velocidad, más resistencia y demás habilidades especiales en función de la raza elegida.

- Se añaden dos zonas de puntuación, una a cada extremo del campo, perteneciendo una a cada equipo. La forma de puntuar consiste en llegar con el jugador que se encuentre en posesión en ese momento del gnomo a los límites de la zona del equipo contrario. De esa forma, el jugador planta el gnomo en el suelo, el equipo al que pertenece obtiene un punto, y un nuevo gnomo es liberado. El equipo con más puntuación al final de tiempo de partida especificado, resulta elegido ganador.

- Se añade un sistema de sprint, mediante el cual un jugador puede aumentar su velocidad durante un tiempo determinado, teniendo que esperar otro tiempo más prolongado para que la acción vuelva a estar disponible.

- Se añade un sistema para que dos integrantes del mismo equipo puedan intercambiar entre ellos la posesión del gnomo.

- Se añade un sistema de puntuación basado en acciones durante la partida. Para que la zona de puntuación esté habilitada para que el equipo portador del gnomo anote un punto al llegar a ella, sus integrantes han de realizar una serie de las siguientes acciones:

    - Golpear al gnomo (sólo realizable por el jugador portador del mismo

    - Completar con éxito un pase al el compañero del equipo

    - Derribar a un jugador rival (sólo realizable por el jugador no portador del gnomo)

Figura 2: Captura de una funcionalidad con la partida final.

3. Evaluación de la idea de negocio

3.1 Estado del arte

3.1.1 Un poco de historia: videojuegos orientados al multijugador local

La modalidad multijugador en los videojuegos ha estado presente prácticamente desde el primer videojuego. Ya en 1972, el conocido juego *Pong*, desarrollado por *Atari*, permitía a dos jugadores competir entre sí. Durante los primeros años de la industria, la tecnología en los hogares no estaba aún lo suficientemente desarrollada ni era fuerte en el mercado, por lo que los videojuegos tenían su mayor público en las máquinas recreativas. Dado que las máquinas se encontraban en locales de juego, era normal que los jugadores asistieran a dichos locales en grupos, por lo que muchas de las máquinas recreativas se orientaban hacia el juego multijugador. *Double Dragon, Street Fighter* o *Teenage Mutant Ninja Turtles: The Arcade Game,* fueron iconos de la industria de los videojuegos durante los años 80 y 90.

Cuando aparecieron las primeras videoconsolas domésticas, se buscaba trasladar la experiencia de juego de las máquinas recreativas al salón del hogar, por lo que muchos de los juegos presentes en las consolas eran directamente versiones de los mismos juegos. Sin embargo, también se comenzó a desarrollar títulos de carácter multijugador diseñados exclusivamente para consolas domésticas y ordenadores personales conforme el mercado iba creciendo, y los videojuegos se asentaban como una forma de entretenimiento más popular.

Los videojuegos de carácter multijugador local buscaban ofrecer la mayor diversión a corto plazo y provocar reacciones de tensión y competitividad en los usuarios, que ayudaran a generar un ambiente de diversión en el espacio real donde se hallaran los jugadores. Destacaban por ser juegos de partidas de corta duración y frenéticos. Es en este tipo videojuegos donde se explotaba el paradigma del diseño de los videojuegos que trata de ofrecer mecánicas fáciles de entender, pero complejas de dominar, para que el juego fuera disfrutable para novatos como desafiante para jugadores habituales.

Los videojuegos más populares de carácter multijugador durante la década de los 90, con las videoconsolas domésticas bien asentadas en el mercado mundial (destacando la *PlayStation* de *Sony* y la *Nintendo 64* de *Nintendo,* con 102 y 38 millones de consolas vendidas, respectivamente*),* se clasificaban principalmente en los géneros que se muestran en la Tabla 3.1:

Género	Máximos representantes
Simulador deportivo	*NBA Live, FIFA*
Acción	*GoldenEye 007, Altered Beast*
Arcade (en referencia a que era el tipo de juego presente en las máquinas recreativas)	*Mario Kart, Bomberman, Bubble Bobble*
Lucha	*Street Fighter, Mortal Kombat*

Tabla 3.1: Referentes de los distintos géneros de videojuegos clásicos

Con la llegada de la llamada sexta generación de videoconsolas, formada por la *Xbox* de *Microsoft*, la *PlayStation* de *Sony*, y la *GameCube* de *Nintendo*, la modalidad orientada al multijugador local pasó a un segundo plano. A pesar de aún contar con una importante presencia en grandes lanzamientos como *Halo: Combat Evolved*, o *Star Wars: Battlefront II*, fue durante este ciclo de videoconsolas cuando se incorporó la vertiente del multijugador a través de Internet. Esta modalidad permitía jugar con un número mayor de jugadores, que además no necesitaban estar en el mismo espacio real o conectados al mismo sistema. El videojuego a través de Internet (*online*), cambió el paradigma de diseño de los modos multijugador en los videojuegos. Si bien anteriormente se precisaba de amigos o familiares para disputar una partida a un juego multijugador, con la inclusión de *Internet*, un jugador podía jugar con otros jugadores reales estando solo. Con ello, el tiempo disponible a invertir en el juego crecía significativamente, y sus sistemas se volvían más complejos, pues la curva de aprendizaje no requería ser tan rápida.

Figura 3: partida multijugador local de Halo: Combat Evolved

La llegada de las videoconsolas de séptima generación (*Xbox 360* de *Microsoft* y *PlayStation 3* de *Sony*) terminaron de asentar el nuevo paradigma. El videojuego con multijugador *online* abarcaba la gran mayoría del mercado, y videojuegos que se basaban en una experiencia para un sólo jugador, como *Bioshock 2* [2], *Mass Effect 3* [3] o *Tomb Raider* [4] se veían obligados a añadir al título un modo multijugador para poder ser competitivos en un mercado donde la competición por Internet reinaba. Los juegos más populares para estas plataformas, *Call Of Duty: Modern Warfare, Gears Of War,* o *Grand Theft Auto V*, resultaban casi imposibles de jugar para aquellos no

familiarizados con el título. Una tarea aún más difícil para un usuario que no sea un habitual consumidor de videojuegos, como un familiar de avanzada edad.
Sin embargo, en pleno auge del multijugador en línea, la compañía nipona *Nintendo* apostó por una videoconsola cuyo pilar principal era el multijugador local. El resultado fue un éxito absoluto de ventas (convirtiéndose en la quinta videoconsola más vendida de la historia) [5]: la *Nintendo Wii*. Con el videojuego *Wii Sports* como principal abanderado del sistema, *Nintendo* buscó revitalizar el videojuego como entretenimiento social, especialmente el familiar. Gracias a los mandos simples e intuitivos y las mecánicas sencillas de sus juegos, prácticamente cualquier persona (sin importar su edad o su experiencia previa con videojuegos), era capaz de disfrutar un juego en el sistema sin apenas dificultad o esfuerzo. Simplificaciones de deportes como el tenis, los bolos o las carreras de karts, conseguían una experiencia sin igual en los comedores de las casas.

Figura 4: Videoconsola Nintendo Wii

Gracias al impacto de la *Nintendo Wii*, el asentamiento de las tiendas de software digitales y las políticas de *Microsoft* y *Sony* por fomentar el desarrollo independiente, el videojuego basado en el multijugador local ha resurgido en el resto de videoconsolas y en los ordenadores personales. Actualmente no resulta especialmente difícil encontrar videojuegos con modos para varios jugadores exclusivamente local con un número de ventas destacable y una buena valoración por parte de los analistas pertenecientes a la prensa especializada. En la table 3.2 se muestran algunos de ellos:

Título	Valoración en Metacritic[8]
Rayman Legends	92 (*Wii U*)
Nidhogg	81 (PC)
Towerfall Ascension	87 (*PlayStation 4*)
Overcooked	81 (PC)
KEEP TALKING AND NOBODY EXPLODES	88 (*Playstation 4)*

Tabla 3.2: Valoración en Metacritic de videojuegos con multijugador local

[8] Plataforma web que muestra la media de todas las valoraciones de la prensa especializada de un título concreto

3.1.2 Plataformas de venta digital

Durante la gran mayoría de la corta historia de la industria de los videojuegos (aproximadamente desde la década de los años 60 hasta hoy), la venta de videojuegos ha ido ligada al formato físico. Dependiendo de la videoconsola en cuestión, el videojuego se incluía en un cartucho (*Atari 2600, Nintendo GameBoy*), *cd-rom* (*Sony Playstation*), o, más recientemente, en formato DVD (*Microsoft Xbox 360*) o Blu-ray (*Sony Playstation 3, Microsoft Xbox One*).

Sin embargo, con el asentamiento de *Internet* en los hogares durante los inicios del siglo XXI, aparecieron dos plataformas de venta digital que cambiarían el futuro de la industria del videojuego. Estas son *Steam*, de la compañía distribuidora americana *Valve,* y *Xbox Live*, de *Microsoft*, conocida mundialmente por el desarrollo del sistema operativo para ordenadores personales *Microsoft Windows.*

Una plataforma de venta digital otorga al usuario la posibilidad de comprar software directamente a través de Internet, accediendo mediante la compra a la descarga de este. Esto no sólo provoca inmediatez, (pues el comprador no ha de desplazarse a una tienda, o esperar el envío del producto de la forma tradicional), sino que además evita los costes de fabricación del soporte físico que contendría el juego (*DVD, Blu-ray*, etc), embalajes, y costes de distribución. Esta consecuencia permite, sobre todo a los pequeños estudios, poder publicar y sacar a la venta videojuegos con mucho menor presupuesto, pudiendo además establecer un precio competitivo en comparación de las grandes producciones. En una tienda digital de videojuegos conviven títulos independientes con precios que suelen comprender un rango entre los 5 y los 30€, y producciones de grandes compañías desarrolladoras, cuyos títulos rondan los 60 - 70€.

Un ejemplo de éxito en los videojuegos independientes gracias a la distribución digital ha sido el título *Minecraft*, desarrollado inicialmente por el desarrollador sueco Markus Persson, que durante 2016 sobrepasó las 100 millones de copias vendidas [6].

A continuación, vamos a describir a dos de las plataformas de venta digital más importantes en el panorama actual.

3.1.2.1 Steam

Steam se presentó durante la Game Developers Conference 9 de 2002, y fue publicado en el año posterior. En un inicio, la plataforma servía como sistema de venta y de soporte para los videojuegos desarrollados por la propia compañía Valve. Fue en 2005 cuando llegó a acuerdos con desarrolladoras externas para vender sus licencias en la plataforma. Actualmente Steam se encuentra en los sistemas operativos para ordenador personal Windows, Mac OS-X y Linux, estando disponible gratuitamente en

[9] http://www.gdconf.com/

237 países. A día de redactar esta memoria Steam cuenta con más de 13 millones de usuarios conectados de forma simultánea, un número aproximado de usuarios totales de 125 millones, y más de 10.000 videojuegos disponibles a la venta (sólo durante 2016 ya se pusieron a la venta 2.245 juegos diferentes). Los beneficios aproximados de Steam durante 2016 fueron de 3.5 billones de dólares. [7]

Las características que hacen a *Steam* ser una plataforma tan popular son la capacidad de tener toda la biblioteca de juegos en un solo lugar, la facilidad que otorga para poder jugar con amigos y los eventos de rebajas (en las rebajas pertenecientes al verano del año 2016, *Steam* generó 223.2 millones con la venta de 36.8 millones de videojuegos)[8].

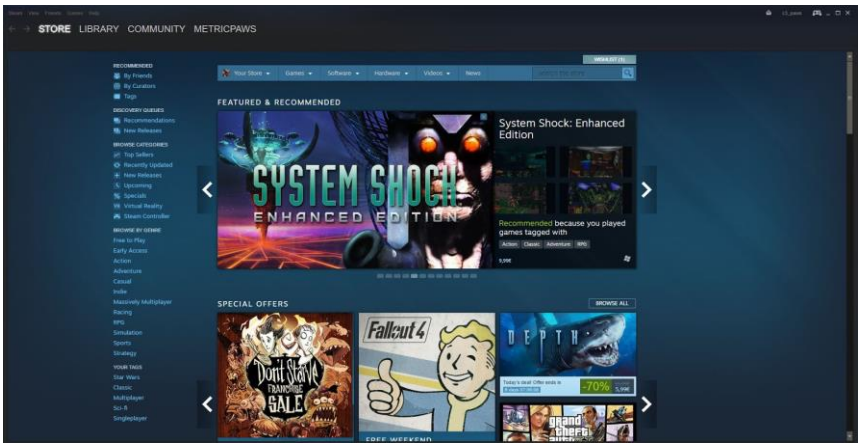

Figura 5: Captura de la interfaz principal de Steam

3.1.2.2 Xbox Live

Xbox Live es la plataforma incluida en las videoconsolas de sobremesa Xbox de la empresa Microsoft. Permite la compra de videojuegos y de contenido adicional para los mismos. Fue con el lanzamiento de su segunda videoconsola de sobremesa, la Xbox 360, cuando Microsoft apostó por el desarrollo independiente mediante la plataforma Xbox Live Arcade. En ella, se publicaron por primera vez grandes juegos de renombre independiente, como Limbo (más de 300.000 copias vendidas en su primer mes en la plataforma), o Braid (título que cuenta con una valoración del 95% en Metacritic, que vendió 55.000 copias durante su primera semana).

Con la reciente salida a la venta de la consola *Xbox One*, *Microsoft* reemplazó la plataforma *Xbox Live Arcade* por una versión todavía en fase temprana llamada *Xbox Live Creators*[10], que permitirá a cualquier desarrollador publicar su videojuego tanto en *Xbox* como en *Windows 10.*

[10] https://developer.microsoft.com/en-us/games/xbox/xboxlive/creator

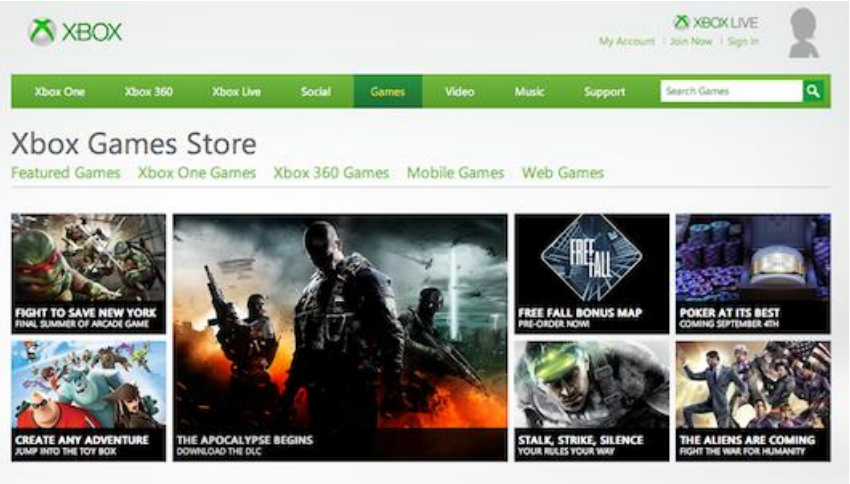

Figura 6: Captura de la interfaz principal de Xbox Live

3.1.3 Videojuegos similares a Gnomore Gnomes

Al comienzo del proceso de desarrollo del prototipo de '*Gnomore Gnomes',* realizamos una búsqueda exhaustiva de títulos orientados a varios jugadores sin conexión a Internet que gozaran de popularidad en la actualidad, y que hubieran sido publicados a lo largo de la última década (2007 - 2017). Después de probar y analizar una decena de juegos, destacamos los siguientes cuatro títulos, ya que recibireron buena aceptación de ventas por parte de los usuarios y que a su vez fueron revisitados con frecuencia a lo largo del desarrollo para estudiar cómo enfocaban algunas características o funcionalidades concretas:

3.1.3.1 Towerfall[11]

Towerfall es un videojuego independiente desarrollado en solitario por Mark Thorson que fue lanzado a la venta en 2014. Sin embargo, a día de hoy, puede encontrarse en las principales plataformas del mercado, incluyendo el ordenador personal, *Playstation 4* y *Xbox One*. Su modelo de negocio consiste en la venta del juego principal tanto en formato físico como en formato digital (con un precio de salida de 14.99$ americanos), y de una expansión de contenidos (con un precio de 9.99$ americanos) que añade al juego principal más escenarios, modos de juego y personajes.

Con una perspectiva en dos dimensiones y un estilo artístico "pixel art" (en homenaje a los videojuegos clásicos), *Towerfall* sumerge a los jugadores en batallas frenéticas y claustrofóbicas para un máximo de cuatro jugadores. Durante dichas batallas, que ocurren en diferentes escenarios con formas de arenas de combate (figura 3.7), la única arma a disposición de los personajes es un arco con contadas flechas. La variedad la aportan los escenarios, pues cada escenario introduce tipos especiales de flechas que afectan a la jugabilidad de la partida, así como una distribución diferente de los obstáculos. Los efectos especiales de las flechas propias de cada escenario

[11] http://www.towerfall-game.com/

incluyen, entre otros, el explotar al impactar, generar zonas tóxicas, o rebotar por las paredes, por lo que cada escenario aporta una jugabilidad radicalmente diferente a pesar de tener la misma base jugable.

Figura 7: Capturas de dos partidas en dos escenarios distintos de Towerfall

Las únicas acciones a disposición del jugador son saltar, disparar y esquivar. Y, sin embargo, debidamente utilizadas, pueden usarse de varias formas para conseguir diferentes resultados. Esa es la gran virtud del título, ya que, en palabras del periodista José Altozano, "su magia reside en que es insultantemente simple, a pesar de que maneja muchas variables y opciones. Un aspecto de su diseño muy agradecido es que absolutamente cualquiera puede entrar y tener una oportunidad."[9]

Towerfall ofrece cuatro diferentes personajes de inicio para elegir, más cuatro personajes adicionales a desbloquear por el jugador. La elección de personaje es puramente estética, pues no tiene efecto en la jugabilidad de la partida. La expansión, llamada "*The Dark World*" incorpora un noveno personaje.

En cuanto a los escenarios, *Towerfall* fue lanzado con siete escenarios. Algunos de ellos se muestran en la figura 7. Más tarde se actualizó de forma gratuita aportando cuatro escenarios más, esta vez teniendo que ser desbloqueados por los jugadores. La expansión, "*The Dark World*", incorpora cuatro escenarios más. La versión para ordenador personal permite a los usuarios construir escenarios personalizados a través de una herramienta incluída en el juego.

En las partidas, diferentes modificadores aparecen de forma aleatoria, habiendo ocho tipos diferentes. Estas modificadores pueden resultar en modificadores de resistencia, velocidad, o alterar el escenario.

Towerfall se puede jugar en dos modos de juego. El primero enfrenta a un mínimo de dos jugadores, hasta un máximo de cuatro. En este modo de juego, que permite personalizar las reglas, los jugadores compiten entre ellos hasta que uno consigue un número determinado de eliminaciones, resultando ganador de la partida. El segundo modo de juego no es competitivo, si no cooperativo. De nuevo, con un mínimo y un máximo de dos y cuatro jugadores, respectivamente, los jugadores se enfrentan a

oleadas de enemigos controlados por la inteligencia artificial. El objetivo en este modo de juego es el de sobrevivir el mayor número posible de oleadas.
Towerfall tuvo una recepción positiva por parte de crítica y público, alcanzando una nota media de 87 en su versión de *PlayStation 4* en la plataforma web *Metacritic*, apareciendo así entre los 10 juegos mejor valorados en *PlayStation 4* durante su año de lanzamiento. Apareció también en la lista de los 50 mejores videojuegos de 2014 [10] elegidos por los lectores de la revista especializada *Eurogamer*.

3.1.3.2 Overcooked[12]

Overcooked es un videojuego multijugador desarrollado por Ghost Town Games y distribuido por *Team17*, lanzado en 2016 para ordenador personal, *PlayStation 4* y Xbox One, con una versión anunciada para *Nintendo Switch*. Originalmente, el juego se vendía únicamente en formato digital. Sin embargo, con el lanzamiento de su única expansión, se lanzó a su vez una versión en formato físico para videoconsola que incluía tanto el juego principal como la expansión.

El juego, de forma alegre y desenfadada, representa el caos de gestionar la cocina de un restaurante. Los jugadores han de enfrentarse con preparar los platos que los clientes van solicitando antes de que se termine el tiempo. Para ello, deberán preparar los ingredientes, cocinarlos y servirlos en el plazo establecido. En función del rendimiento de los jugadores durante el tiempo que dura la partida, se les galardona con estrellas, siendo cero estrellas la peor puntuación, y tres la máxima.

Figura 8: Capturas de dos partidas en dos escenarios distintos de Overcooked

El juego principal ofrece treinta escenarios en los que jugar. Estos se disponen en serie y, para acceder a uno de ellos, se debe haber completado todos los anteriores. La expansión de contenido incluye seis escenarios adicionales.

Los personajes a elegir afectan únicamente en el campo estético, ya que todos los personajes tienen las mismas habilidades. La expansión incluye seis personajes más.

La variedad y rejugabilidad se consiguen gracias a los escenarios, ya que varían las recetas a preparar por los jugadores (como se muestra en la parte superior de la figura 8), e incluyen elementos que afectan a la jugabilidad. Como ejemplo, un escenario con el suelo hecho completamente de hielo, dificulta el control de los cocineros para poder servir o manipular los ingredientes. Su jugabilidad destaca por ser extremadamente simple.

[12] http://www.ghosttowngames.com/overcooked/

Las únicas acciones permitidas son:

- Mover al personaje.
- Coger un objeto, como un ingrediente, plato o sartén.
- Usar un objeto, que puede ser desde cortar un ingrediente o fregar un plato, a comenzar a hervir o freír un ingrediente.

La simpleza de las acciones consiguen una curva de aprendizaje mínima. En cuestión de segundos, un jugador nuevo puede entender y jugar sin dificultad en los escenarios más básicos. La dificultad se encuentra a la hora de organizarse las tareas entre los jugadores, creando situaciones desternillantes entre los jugadores, que van pidiendose a gritos que les traigan nuevos ingredientes o que sirvan los platos ya cocinados. Por ello es un juego que no se entiende fuera del multijugador local.

Overcooked alcanzó una nota media de 81 en la página web *Metacritic*. Fue ganador del premio a mejor debut en los *Industry Game Awards,* nominado al premio Mejor Juego Multijugador en los *Game Awards* del año 2016, así como al premio a excelencia en el diseño en el *Independent Games Festival,* entre otras nominaciones y premios.

3.1.3.3 Mario Strikers: Charged[13]

Mario Strikers: Charged es un videojuego desarrollado por el estudio *Next Level Games* y distribuido por *Nintendo*, lanzado para la videoconsola de sobremesa *Nintendo Wii* el 25 de mayo de 2007. Posteriormente fue publicado en la consola virtual *Nintendo Wii U*, videoconsola sucesora de la Wii original, en el año 2016. Es un juego de carácter *arcade* deportivo orientado tanto al multijugador local como al multijugador a través de Internet, para un máximo de 4 jugadores. En él, los personajes más conocidos de las diversas franquicias de la compañía de videojuegos japonesa *Nintendo*, se disputan la victoria en un partido de fútbol. Las reglas y mecánicas caóticas que se añaden a las bases del fútbol, convierte a los estadios en verdaderos campos de batalla.

A pesar de que la práctica de publicar contenido descargable adicional para los videojuegos que ya se encontraban a la venta empezaba a asentarse en la industria durante 2007, *Nintendo* mantuvo su política de videojuegos sin contenido adicional[11] de pago hasta que la adoptó en sus videojuegos a partir del año 2014 en títulos como *Mario Kart 8*. Por tanto, *Mario Strikers: Charged*, anterior, se vendía como un producto de un único pago y todo el contenido adicional disponible era desbloqueable desde el propio juego, obteniéndose este a través de desafíos. El precio de salida fue de 39.99$ americanos.

[13] https://microsite.nintendo-europe.com/mariostrikers/

El juego se presenta con tres modos de juego:

- **Dominación**: Modo de juego en el cual el rival es controlado por inteligencia artificial, es decir, no por un jugador real. Este juego permite alterar algunas reglas de la partida, como las condiciones de victoria o la duración de la misma.

- **Copa Goleadores**: El jugador participa en 3 torneos, eligiendo previamente el capitán deseado y la dificultad. Los participantes deben superar una serie de rondas clasificatorias y eliminatorias. Al finalizar los torneos se otorgan los trofeos al equipo menos goleado y al jugador más goleador.

- **Situaciones Límite**: Es un modo de juego consistente en una serie de desafíos que, al ser completados, otorgan recompensas y elementos desbloqueables para el jugador.

Las partidas pueden disputarse en un conjunto total de diecisiete estadios:

- Cuatro estadios iniciales (Dos de ellos se muestran en la figura 9).

- Seis estadios desbloqueables.

- Siete estadios clásicos de ediciones del videojuego anteriores.

Los estadios desbloqueables se obtienen tras completar torneos concretos, y/o consiguiendo el máximo número de goles durante los mismos.

Figura 9: Capturas de dos partidas en dos escenarios distintos de Mario Strikers: Charged

Los personajes elegibles por los usuarios comprenden un total de doce opciones. De ellos, ocho se encuentran disponibles desde el inicio del juego, y cuatro son desbloqueados para el jugador cuando éste completa ciertos torneos. Cada uno de ellos posee una serie de habilidades especiales y tiene la capacidad de poder usar objetos para inclinar la balanza del partido a su favor, como, por ejemplo, bombas, trampas, elementos que aumentan la fuerza, la velocidad, etc. Esto amplía las posibilidades de la partida, pues expande los conceptos básicos que presenta el fútbol, y permite al jugador experimentar con diferentes estrategias.

La idea jugable es sencilla: el equipo con el mayor número de goles en su marcador antes de que se termine el tiempo (o llegue a una determinada cantidad de goles marcados, dependiendo del modo de juego), se lleva la victoria. Otra diferencia que destaca frente al resto de videojuegos deportivos, es el balón. El comportamiento de este se ve influído por las acciones del jugador, ya que, conforme se vayan realizando acciones (tales como pases, lanzamientos o habilidades especiales) el balón brilla de forma gradual, lo que afecta a la fuerza, velocidad y precisión con la que puede ser lanzado. Al igual que en *Mario Strikers: Charged*, *Gnomore Gnomes* posee la característica de que el balón tiene un comportamiento único. En nuestro caso, el gran valor añadido es que éste se trata de un ser vivo, concretamente de un gnomo. Esto genera un cambio en las mecánicas básicas, pues los jugadores deben perseguirlo por el terreno de juego para poder hacerse con él o evitar que los jugadores contrarios lo consigan.

En conclusión, algunas de las características mejor valoradas en este título deportivo fueron la inclusión del modo multijugador online y el uso de un control más clásico que resultaba intuitivo y fácil de dominar. Debido en parte a estas características y a la cantidad y variedad de su contenido, el juego obtuvo buenas críticas, obteniendo una nota de 79 sobre 100 en *Metracritic* (basado en 47 análisis) y una muy buena aceptación por parte de los jugadores, obteniendo un 8.1 sobre 10 de nota media por parte de 127 análisis realizados por usuarios del juego. Prueba de ello es que el juego ocupa el puesto 11 de la lista de los videojuegos *Nintendo Wii* mejor valorados del año 2007. Debido a estos motivos y a la similitud en ciertos aspectos esenciales con la idea que buscábamos obtener en *Gnomore Gnomes*, fue un título que tuvimos muy en cuenta a la hora buscar inspiración.

3.1.3.4 Stikbold![14]

Stikbold! es un videojuego desarrollado por el estudio *Game Swing* que fue lanzado para las plataformas online *Playstation Network, Xbox Live y Steam* el 5 de abril de 2016. Es un juego de carácter *arcade*/deportivo orientado al juego individual o al multijugador exclusivamente local **(**excluyendo el modo a través de Internet) en el que pueden participar hasta un máximo de 6 jugadores simultáneos controlados bien por la computadora o por otros usuarios. En *Stikbold!,* los jugadores participan en una representación de una competición del deporte conocido en américa como *dodgeball,* o más conocido en nuestro país como balón prisionero. En ella, los usuarios deben lanzarse una pelota para eliminar del terreno de juego a los contrarios y así conseguir la victoria. Los distintos escenarios aportan leves cambios en la jugabilidad básica del juego otorgando variedad al producto y aumentando su rejugabilidad.

Stikbold! posee un modelo de negocio de producto de único pago por el valor de 9,99€ a través de las plataformas online mencionadas anteriormente sin poseer ningún tipo de contenido adicional ni expansión de pago, dando la opción de adquirir, eso sí, su banda sonora de forma totalmente gratuita. El título recibió un nuevo nivel jugable gratuito que completaba los ya existentes en su versión básica. Todo el restante

[14] http://www.stikbold.com/

contenido adicional presente en el juego puede desbloquearse jugando y nos permite obtener nuevos personajes.

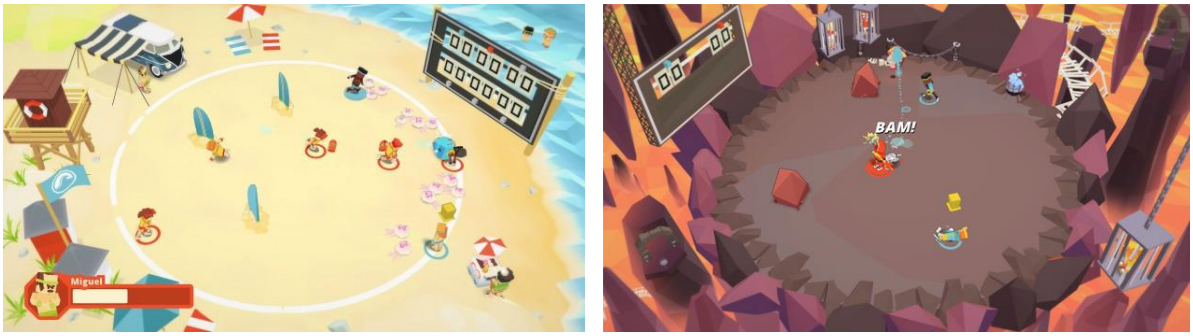

Figura 10: Capturas de dos partidas en dos escenarios distintos de Stikbold!

El videojuego nos da la opción de escoger entre varios modos de juego, siendo estos los siguientes:

- **Modo Historia:** en el cual el jugador avanza a través de una aventura, compitiendo en diversos partidos de balón prisionero para avanzar por los distintos niveles. Hay que destacar que este modo de juego está enfocado únicamente a un sólo jugador y los rivales son controlados por una inteligencia artificial.

- **Equipo VS Equipo:** en este modo de juego se da a elegir entre la competición contra otros jugadores reales, o contra la inteligencia artificial. Se forman dos o tres equipos (diferenciados por el color de las equipaciones), que deben competir en un partido.

- **Todo vale:** es el clásico modo de juego conocido también como *Deathmatch* o como "todos contra todos". En este modo, cada jugador selecciona el personaje que desee de entre todos los que ha conseguido desbloquear y pugna con el resto de jugadores por conseguir la cantidad de puntos indicados en las reglas propias de cada uno de los escenarios.

- **Minijuegos:** a parte de todos los modos nombrados anteriormente, *Stickbold!* También posee una serie de minijuegos donde el concepto jugable varía completamente, siendo una experiencia menos profunda que el modo de juego básico pero que complementa al mismo ofreciendo más posibilidades para evitar la monotonía de los modos de juego anteriores. Entre ellos podemos encontrar los siguientes:

    - **Balonmano**: la gran diferencia de este minijuego respecto al modo de juego básico es que cada equipo posee una portería propia como si de un juego de fútbol o balonmano se tratase. La forma de puntuar cambia siendo ahora el clásico sistema de goles el que ayudará a los equipos a conseguir la victoria. Este minijuego está enfocado únicamente para 2 o

4 jugadores humanos, es decir, los equipos no pueden complementarse con personajes controlados por el ordenador.

- **Pop Pop**: en este minijuego cada jugador tiene una pelota de playa atada a uno de sus pies. El jugador alcanza la victoria evitando que desinflen su pelota y desinflando la de los demás. El escenario en el que tiene lugar este minijuego posee varios obstáculos que dificultan y otorgan cierta estrategia a la partida. De la misma manera que el minijuego anterior, Pop Pop está enfocado para 2 o 4 jugadores controlados únicamente por jugadores reales.

- **El gran escape:** en el gran escape cada jugador es poseedor de unas cajas de madera tintadas del color la equipación del personaje. La victoria en este minijuego se consigue al conseguir defender tus propias cajas y al destruir las cajas del contrario. Al igual que los otros dos minijuegos anteriores, el gran escape está enfocado para 2 o 4 jugadores controlados únicamente por jugadores reales.

- **Arcoiris a tope:** este minijuego extiende el modo de juego básico, ya que la manera de ganar es la misma, salvo que el escenario está diseñado con la intención de generar el mayor caos posible. La cantidad de obstáculos presente en él es enorme, entorpeciendo así la experiencia jugable para los usuarios. Al igual que los minijuegos anteriores, el gran escape está enfocado para 2 o 4 jugadores controlados únicamente por jugadores reales.

Los diferentes minijuegos y niveles poseen particularidades no sólo estéticas, sino que además aportan características únicas para cada uno de ellos que interfieren en la jugabilidad de la partida. De esta manera *Stikbold!* aumenta la rejugabilidad evitando que el jugador caiga en la monotonía. Además ofrece diez personajes desbloqueables para complementar a los ocho iniciales.

El juego, por lo general, obtuvo buenas críticas en la prensa especializada, logrando una valoración de 75 sobre 100 en la página web *Metacritic*. En la valoración por parte de los usuarios, este título obtuvo una nota media de 79 sobre 100. Uno de los puntos más comentados de *Stikbold!* es su intuitivo manejo, convirtiendo al título en un juego sencillo de controlar pero que exige más dedicación si se quiere dominar, sobre todo en sus niveles de dificultad más altos. Su estética humorística basada en los años 70 lo hace muy atractivo (teniendo en cuenta la simplicidad de la misma). Pero, sin duda, una de sus mayores bazas es su relación calidad/precio, ya que, sin contar con un gran número de niveles ni de modos de juego alternativos, sus 9.99€ de precio lo hacen un producto muy recomendable para pasar buenos ratos entre amigos y familiares.

3.1.3.5 Resumen

Los videojuegos anteriormente analizados nos permitieron estudiar qué características o funcionalidades eran necesarias incluir, cuáles podrían resultar interesantes, y cuáles eran prescindibles o innecesarias. Como hemos indicado a lo largo de los análisis anteriores, la inclusión de escenarios interactivos y diferenciados, era clave para conseguir un producto sólido, variado y rejugable.

A continuación, en la Tabla 3.3 se resumen las características presentes o ausentes en los cuatro títulos analizados, y si estas están incluidas o no en 'Gnomore Gnomes'.

	TowerFall Ascension	Overcooked!	Super Mario Strikers	Stikbold!	***Gnomore Gnomes***
▪ **Tutorial**	✗	✓	✓	✓	✓
▪ **Modo para un jugador**	✗	✗	✗	✓	✗
▪ **Modo Multijugador Online**	✗	✗	✓	✗	✗
▪ **Modo Multijugador local**	✓	✓	✓	✓	✓
▪ **Número de jugadores**	1-4	1-4	1-4	1-6	1-4
▪ **Número de escenarios iniciales**	7	30	4	6	4
▪ **Número de escenarios desbloqueables**	4	0	6	0	3
▪ **Número de modos de juego**	2	2	3	5	2
▪ **Diferencias jugables entre personajes**	✗	✗	✓	✗	✓
▪ **Elementos estéticos desbloqueables**	✗	✗	✗	✗	✓
▪ **Sistema de logros**	✓	✓	✗	✓	✓

					***Gnomore Gnomes***
▪ **Sistema de *Ranking***	✗	✗	✗	✗	✗
▪ **Sistema de perfiles de usuario**	✗	✗	✗	✗	✗
▪ **Soporte para *Steam Workshop***	✓	✗	✗	✗	✗
▪ **Contenido adicional de pago**	✓	✓	✗	✗	✓
▪ **Plataformas**	*Steam,Xbox Live, Playstation Network*	*Steam,Xbox Live, Playstation Network*	*Nintendo Wii/Wii U*	*Steam,Xbox Live, Playstation Network*	*Steam, Xbox Live*
▪ **Precio de salida**	14,99$	16,99$	19,99$	9,99$	9,99$

Tabla 3.3: Tabla comparativa de los videojuegos similares a "Gnomore Gnomes"

Como podemos observar, una de las grandes diferencias que aportaría *Gnomore Gnomes* sería su contenido desbloqueable. Tenemos pensando elaborar una gran cantidad de accesorios y aspectos distintos para cada uno de los personajes seleccionables del videojuego. Éstos tendrían un precio muy reducido y saldrían a la venta en forma de *packs* que podrían ser incluso temáticos. Otra de las virtudes de nuestro videojuego sería la jugabilidad única que aportaría cada personaje, ya que muy pocos juegos de este género en pc aportan esa característica. Por otra parte, decidimos elegir ese precio tan ajustado para el producto base, porque al ser nuestro primer videojuego en venta y no ser conocidos, pretendemos llegar a la mayor cantidad de público posible y de esta manera esperamos conseguir ese propósito. La manera de conseguir beneficios extra al precio del producto base, sería a partir de pequeñas expansiones del propio juego o de la venta de elementos personalizables como se ha mencionado anteriormente.

3.2 Lean Canvas

El Lean Canvas facilita la definición de las diversas partes de una propuesta de proyecto de cara a su presentación en una *startup.* En él se muestran, entre otros, los puntos fuertes, las debilidades, los costes, la forma de ingresos, o el mercado al cual va destinado. La siguiente tabla muestra el lean canvas que realizamos el primer día que nos embarcamos con el proyecto:

2 **Problema**	**4** **Solución**	**3** **Proposición de valor**	**9** **Ventaja competitiva**	**1** **Clientes**
• La industria del videojuego se encuentra estancada en la repetición de las mismas fórmulas, tratando de replicar el éxito de juegos populares, perdiendo así el interés del jugador clásico que busca experiencias innovadoras. • La gran mayoría de los videojuegos multijugador que aparecen en el mercado están enfocados al llamado deporte electrónico, dejando de lado las experiencias que buscan la diversión sencilla entre amigos, sin exigir horas de dedicación hasta dominarlo.	• Videojuego con mínima curva de aprendizaje, accesible para cualquier tipo de jugador • Mecánicas jugables únicas que no se encuentran en ningún otro título a la venta, diseñadas específicamente para el juego entre amigos / familiares.	Cuando extraes lo mejor del *rugby,* del balón prisionero, cambias el balón por un gnomo que desesperadamente trata de huir de los jugadores, todo ello concentrado en una serie de estadios locos en un mundo de fantasía, aparece el videojuego deportivo más frenético y descontrolado que más amistades dinamita.	El coste de desarrollo del proyecto es significativamente inferior al coste estándar de desarrollo de un videojuego de sus características, permitiendo un precio de venta inferior a los juegos de la competencia.	El cliente objetivo a quien va dirigido el producto es el usuario de videojuegos habitual o casual, con un rango de edad de 8 años aproximadamente hacia delante, con afición por los videojuegos de carácter multijugador sin conexión a Internet. Los early adopters son los miembros de la asociación de videojuegos e-Sports Valencia y los participantes de eventos de testeo de videojuegos.

7 Costes	8 Métricas		5 Canales	6 Ingresos
• Horas de trabajo (aproximadamente 300h): 19.200€ • Actualización equipos: 1100€	• *Feedback* en eventos de testeo de videojuegos • Nº de visitas y valoración del trailer en *Youtube* • Donación total en la campaña de financiación • Nº de descargas de la demo • Ventas en *Steam* • Jugadores/Tiempo en *Steam*		• Los canales de comunicación serán *Facebook*, *Youtube* y *KickStarter*. • Los canales de distribución serán *Steam*, con la posibilidad de distribución en *Xbox Live*	El modelo de negocio para el producto va a ser la venta del producto mediante distribución digital en la plataforma *Steam, con un precio de 10€ por licencia*, ofreciendo además expansiones de contenido para el mismo por un coste adicional menor al del producto original (5€ por cada paquete de contenidos).

3.3 Análisis DAFO

El análisis DAFO es una herramienta que resume la situación de un proyecto, mostrando las principales características propias del proyecto (fortalezas y debilidades), así como su situación frente al mercado al que se enfrenta (oportunidades y amenazas). A continuación se muestra el análisis que se realizó durante los primeros días del proceso de desarrollo:

FORTALEZAS	**DEBILIDADES**
• Precio competitivo • Mecánicas jugables únicas • Diseño original de personajes • Diseño innovador de escenarios • Bajo coste de desarrollo • Enfocado a amplio rango de edades • Sin limitación geográfica • Curva de aprendizaje mínima • Variedad en los roles jugables	• Escaso contenido inicial • Marketing efectivo • Presencia online • Estado Financiero • Primer producto a la venta
OPORTUNIDADES	**AMENAZAS**
• Escasas propuestas similares en el mercado • Revitalización de propuestas basadas en el multijugador local	• Saturación en el mercado de videojuegos • Competencia muy asentada

Tabla 3.0-1: Análisis DAFO del proyecto

El esfuerzo que hemos realizado para el diseño de personajes, escenarios y mecánicas es nuestra gran fortaleza. La creatividad detrás de cada elemento en el videojuego le hacen destacar frente al resto de competidores que se limitan a clonar las fórmulas y elementos exitosos. Adicionalmente, el bajo coste de desarrollo del proyecto (en comparación a los costes estándar de desarrollar un videojuego), permite un precio de venta más que competitivo.

Por otro lado, nuestras debilidades se centran en que dependemos de la campaña de *crowdfunding* para cubrir los costes, por bajos que estos sean, y que, al ser nuestro primer videojuego, carecemos de experiencia en el mercado, de reputación, y de presencia online que nos permita realizar un marketing sólido y efectivo.

En cuanto al análisis de oportunidades y amenazas, nuestra mayor oportunidad pasa por aprovechar la creciente revitalización del género del multijugador local. Sin embargo, estudios indican que la facilidad para publicar videojuegos en estos últimos años está llevando a la industria a un estado de saturación. Prueba de ello es que el 40% de los videojuegos publicados en la plataforma *Steam*, fueron lanzados durante el año 2016. Esto supone una amenaza seria para nuestro proyecto, debido a las debilidades anteriormente indicadas.

3.4 Proyección económica

Una de las partes más importantes a la hora de afrontar el desarrollo de un proyecto es la definición de la proyección económica, que estima la relación entre costes e ingresos a lo largo del tiempo de desarrollo y de la vida del producto terminado una vez se encuentre a la venta. Con estas estimaciones se puede estudiar la viabilidad de la idea, estableciendo una aproximación de en qué momento el producto comenzará a aportar beneficios, o cómo se espera que evolucione el número de clientes.

En la tabla siguiente se muestra el cálculo de la proyección económica que se realizó durante los primeros compases del proyecto. Se dividieron los intervalos de tiempo por años, estudiando la evolución durante los primeros 5, pues a partir del tercer año ya no deberían aparecer costes significativos, lo cual debería ser representativo de los beneficios que aportará el videojuego posteriormente. Los parámetros que se analizan son: número de nuevos compradores del videojuego, nuevos compradores de los contenidos adicionales del mismo, ingresos obtenidos, costes y beneficios.

Como se indica en el Lean Canvas, el precio elegido para el producto es de 10€. Este precio se estableció tras un estudio de los precios de los juegos contra los que competiría, siendo este un poco inferior a la media, dada la poca reputación del estudio y que no es un videojuego con una carga de contenido demasiado elevada en su versión básica. Esa versión podría ser ampliada con paquetes de contenido adicional, con un precio reducido (concretamente la mitad del precio de la licencia básica, 5€).

	Año 1	Año 2	Año 3	Año 4	Año 5
Nuevos compradores					
Licencia del videojuego	-	200	500	500	400
Licencia rebajada del videojuego	-	-	-	-	1000
Contenido adicional	-	-	100	150	200
Ingresos					
Licencia del videojuego	-	2000	2000	5000	4500
Contenido adicional	-	-	500	750	1000
Costos					
Personal	1.000 €	1.000 €	-	-	-
Equipos	200 €	-	1.000 €	-	-
Beneficios (Ingresos - Gastos)	-1.200 €	-200 €	1.300 €	7.050 €	12.550 €

Tabla 3.4: Proyección económica para los primeros cinco años

La estimación se ha realizado teniendo en cuenta que el producto base tendrá un valor de 9,99€ frente a los 5€ que valdría durante las rebajas de la plataforma de venta digital. Por otra parte, el contenido adicional a modo de expansiones para el videojuego también tendría un valor de 5€ para cada una de ellas.

Dado que hasta unos años el juego no generaría beneficios y que se trata de nuestro primer proyecto debemos ser cautos. No nos planteamos la posibilidad de que el videojuego sea nuestra principal fuente de ingresos, por lo que los beneficios obtenidos por el mismo serían un ingreso extra complementario a otros empleos. Para cubrir las pérdidas de los dos primeros años optamos por la opción de publicar una campaña de crowdfunding en la plataforma *kickstarter*, donde esperamos recaudar suficientes donaciones.

A partir del cuarto año, se espera un incremento de usuarios notable. Si se alcanzara un número de jugadores significativo (aproximadamente 5.000 jugadores), se estudiaría tanto lanzar el videojuego en otras plataformas, como añadir más contenido adicional. Pero, llegados a ese caso, se debería tratar como una proyección económica nueva, ya que supondría nuevos costes, y, por tanto, de necesaria financiación. En este proyecto se ha analizado la proyección económica con la idea de lanzar el juego en una sola plataforma con un sólo paquete de contenidos adicionales.

4. Proceso de desarrollo

El desarrollo de un videojuego es un proceso largo y complejo. Hasta los videojuegos en apariencia más simples conllevan un importante trabajo de diseño e implementación detrás. Cualquier mínimo fallo en la programación, o decisión de diseño imperfecta puede arruinar la experiencia jugable por completo. Para entender mejor esta cuestión, se va a tratar de ejemplificar mediante un error de programación que se produjo durante el desarrollo de *Gnomore Gnomes* que muestra cómo la atención al detalle y la organización es capital para alcanzar un producto final robusto, en los plazos previstos.

El error en cuestión se dió lugar en la acción que realizaba el lanzamiento del gnomo. El jugador que tiene el gnomo en posesión, puede lanzarlo hacia delante varios metros si así lo desea. El gnomo porta un sistema de detección de colisiones que permite que, al detectar la colisión con un obstáculo (muro del escenario u otro personaje), choque con este y así se detenga el lanzamiento. Durante el proceso de implementación se completó el código y se dejó la acción lista para realizar las pruebas. Durante ellas, se lanzó el gnomo de todas las maneras posibles, obteniendo un resultado satisfactorio. Sin embargo, unos días después, durante unas pruebas más generales de la partida, tras un lanzamiento el gnomo desapareció de los límites del terreno de juego. Esto dejaba injugable la partida, pues no había ningún sistema que controlara ese tipo de errores, y sin gnomo en el terreno de juego, la partida no puede continuar.

El problema, que no habíamos previsto, era el siguiente. La animación de lanzamiento alarga el brazo del personaje hacia delante, similar al movimiento de lanzamiento que realiza un jugador de *baseball.* Esto provocaba que, si el personaje se encontraba pegado a los límites del campo cuando el brazo estaba extendido, el gnomo (que se encuentra en ese momento en la mano del personaje) ya se encontrara fuera de los límites del campo, por lo que no se produce la detección de la colisión antes comentada, y terminaba precipitándose hacia el infinito. Esto demuestra cómo la acción más simple puede desmoronar la jugabilidad de la partida, y cómo el proceso de pruebas ha de ser más exhaustivo y profundo que en otro tipo de software.

Este tipo de situaciones puede descontrolarse si no se lleva el desarrollo del proyecto de forma metódica y planeada, definiendo qué funcionalidades se están desarrollando, cuáles están completadas, cuáles en proceso de pruebas, etc. Otro aspecto clave en la organización del desarrollo de un producto software es la gestión de las versiones. Esto comprende decidir qué funcionalidad estará incluida en cada versión, y en qué fecha estará completa cada una de ellas.

De todas las versiones por las que pasa un producto software, una de las más importantes es el llamado Mínimo Producto Viable (MVP), que corresponde a la versión que ya puede ponerse a la venta, aunque en ese momento no incluya toda la funcionalidad que portará la versión final. Por ello, en las primeras etapas del proyecto, se definió el mapa de características, que incluye el global de funcionalidades que se incluirá en el producto, y que permite señalar cuáles de ellas estarán incluidas en el

Mínimo Producto Viable, y cuáles se incorporarán más adelante. A continuación se muestra el mapa de características que definimos para el proyecto *Gnomore Gnomes.*

4.1 Mapa de características

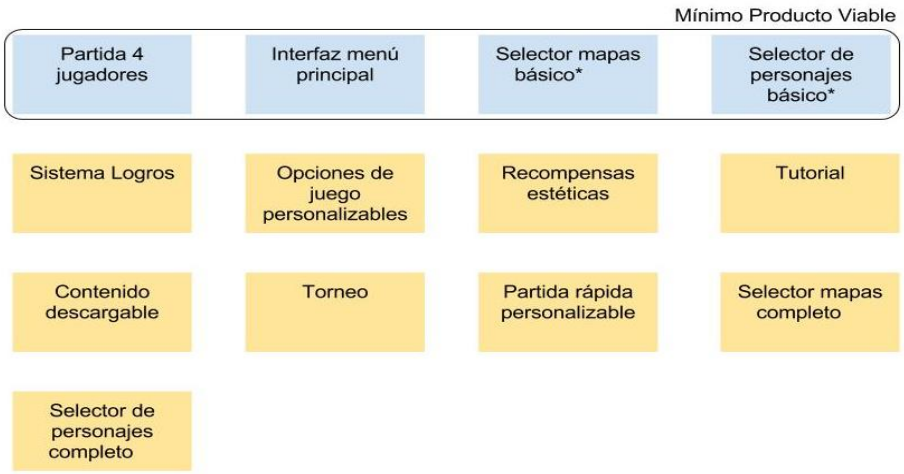

Figura 11: Mapa de características de *Gnomore Gnomes*

Una vez está definido el mapa de características, se estudia cuáles de ellas se incluirán en el Mínimo Producto Viable. Analizando los recursos y tiempo disponibles, se llegó a la conclusión de que incluiría las cuatro características marcadas en el mapa:

- Partida 4 jugadores:

Permite a cuatro jugadores competir en una partida de 5 minutos de duración repartidos en dos equipos de dos jugadores. Al final del tiempo reglamentado, se muestra qué equipo ha resultado ganador, para después volver al menú principal.

- Interfaz menú principal :

En el menú principal del MVP, se ofrece al usuario jugar una partida, entrar en las opciones, o salir del software. Un modelo de un gnomo en la parte izquierda de la pantalla indica qué opción está seleccionada. Un detalle interesante sobre el menú principal es que las opciones caen desde la parte superior de la pantalla usando el sistema de físicas en tiempo real, dando la sensación de que son bloques con peso que rebotan contra el suelo al caer.

- Selector de escenario básico

El selector de escenario básico permite al jugador seleccionar el escenario en el que se va a desarrollar la partida, por lo que es la pantalla posterior a elegir la opción 'Play' en el menú principal. En la pantalla se muestran representaciones simplificadas de los diversos escenarios disponibles para elegir. Cuando el usuario selecciona el escenario

en el que quiere jugar la partida, aparece la pantalla de selección de personajes básica.

- Selector de personajes básico

El selector de personajes incluido en el mínimo producto viable es la parte del videojuego donde cada jugador elige qué raza quiere para su personaje, así como elegir a cuál de los dos equipos disponibles quiere pertenecer (siempre que este no esté ya formado por dos jugadores). En la versión completa de esta característica, disponible en la versión final, el jugador podrá personalizar a su personaje con adornos obtenidos a los largo de las partidas (como por ejemplo cascos, trajes o peinados), así como cambiar el color base del equipo. Una vez todos los jugadores han seleccionado la raza de su personaje, la partida para cuatro jugadores da comienzo.

4.2 Cronología del proyecto

Una vez definidas todas las características, y con el desarrollo del mínimo producto viable como objetivo, había que planear el proceso que nos llevaría hasta ello. La fecha que se estableció para tener el primer prototipo que pudiera ser probado con *early-adopters* fue enero. Encontrándonos en octubre cuando comenzamos el proyecto, era un intervalo demasiado amplio como para no subdividir el prototipo en pequeños módulos jugables que fueran integrándose conforme se desarrollaran, y así tener plazos más reducidos a lo largo de los cuatro meses. Para que la gestión de las tareas pendientes y de aquellas por completar fuera cómoda, útil y fácil de compartir, se empleó la plataforma *Trello.* En ella, definimos en forma de fichas (figura 12) las distintas tareas a realizar, que íbamos moviendo entre distintas secciones, y representaban funcionalidades del videojuego:

- To Do: En esta sección se encontraban las tareas que estaban pendientes de realizar. Se encontraban englobadas en dos tareas principales: “Partida mínima”, que indicaba lo que había de incluir el prototipo, y “Partida completa”, que indicaban aquellas características a implementar a partir de enero, al tener completo el prototipo.

- Design: Aquí se encontraban las tareas de tareas que se encontraban en fase de diseño. Esta es la fase en la cual se da forma a la funcionalidad perteneciente a la tarea, decidiendo en qué consiste y cómo ha de funcionar. Era muy común retornar acciones a esta fase cuando, durante las pruebas, comprobábamos que había problemas de diseño con alguna de dichas funcionalidades.

- Doing: En esta sección aparecían las tareas de tareas que se encontraban implementándose en ese momento. Diferenciamos la sección en dos: Una para la representación artística de la funcionalidad, y otra para la programación de la misma.

- Test: Cuando se hacía una versión de las implementaciones, las tareas pasaban a la sección Test, en la que hacíamos pruebas internas, comprobando que el funcionamiento fuera el deseado. Siendo un juego para cuatro jugadores y nosotros dos desarrolladores, muchas de esas pruebas eran complicadas de llevar a cabo, y no era hasta que realizábamos las pruebas con algunos amigos voluntarios, que no podíamos comprobar si la funcionalidad funcionaba correctamente en cualquier situación.

- Done: Aquí terminaban las tareas que recibían la aprobación tras los períodos de prueba.

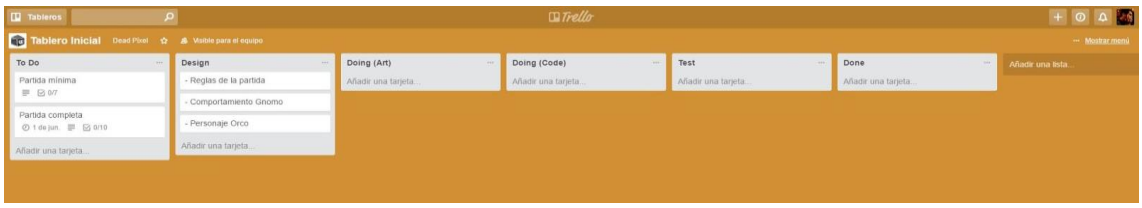

Figura 12: Captura de la ficha perteneciente a la tarea de desarrollo del personaje

Cuando iniciamos el desarrollo en Octubre, definimos las dos tareas de “Partida mínima” y “Partida Completa”, y colocamos las primeras tareas en la etapa de diseño. Estas fueron las reglas básicas de una partida, el comportamiento del gnomo en el terreno de juego, y el funcionamiento de un personaje (para el primer prototipo, los personajes estarían representados todos como Orcos). Como se muestra en la siguiente captura, la tarea de “Personaje Orco” comprendía todas las acciones a realizar por un personaje.

La captura mostrada a continuación pertenece al 10 de Noviembre, cuando la tarea se encontraba en la sección “Doing (Art)”, en la cual se estaban implementando las animaciones de cada una de las acciones. Cuando las animaciones del personaje estaban completadas, se pasaba a gestionarlas a través del código en la etapa “Doing (Code)”, activando o desactivando las animaciones en función de lo que sucediera en el videojuego. El proceso de animación y de programación se muestra en profundidad en el capítulo 6 de las distintas memorias.

Figura 13: Captura de la ficha perteneciente a la tarea de desarrollo del personaje

El desarrollo avanzaba según lo previsto, hasta que, probando una de las acciones del personaje, esta afectaba al funcionamiento del resto, dando lugar a situaciones indeseadas en la jugabilidad. Más concretamente, era la acción de *sprint* la que provocaba colisiones con el resto de acciones. Aunque se profundiza más en esta acción más adelante en este capítulo, basta con decir que era una acción que proporcionaba un instante de aceleración en la velocidad de movimiento del personaje. Esta acción producía problemas al ser combinadas con las acciones de coger el gnomo o lanzar el mismo. Se debía a que el código perteneciente al personaje no estaba preparado para gestionar dos acciones simultáneamente, como eran, por ejemplo, realizar un *sprint* a la vez que se trata de coger el gnomo.

No hubo otra solución que replantear y reescribir todo el código del personaje, suponiendo un duro revés con los plazos previstos (como se muestra en la figura 14). El nuevo sistema de control de acciones de los personajes no estuvo listo hasta un mes después, a mediados de diciembre, lo que obligó a retrasar la prueba con *early-adopters* de principios de enero a finales del mismo.

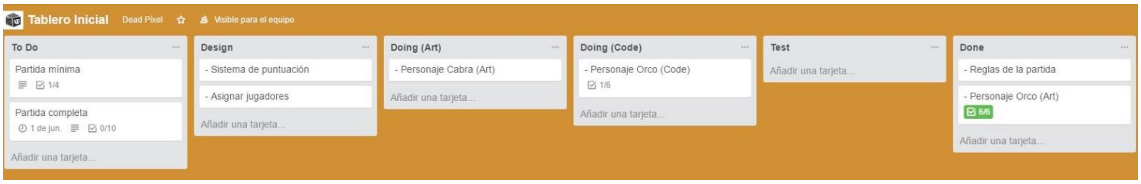

Figura 14: Captura del tablero de tareas del 19 de diciembre

En la captura anterior se muestra el estado de las tareas, mientras se corregía el problema con el manejador de acciones del personaje. Como se puede observar, las animaciones del personaje se encontraban completas, y ya se empezaba a trabajar en las animaciones del siguiente tipo de personaje. A su vez, nuevas funcionalidades se encontraban en la etapa de diseño, y la programación del primer personaje permanecía en la sección “Doing”.

Llegado el final del mes de enero de 2017, conseguimos, según los plazos que nos habíamos establecido, tener lista una primera versión jugable con los aspectos básicos del juego en la que poder probar una partida completa entre cuatro jugadores. Como dato curioso, el tiempo empleado en corregir el código había permitido que las animaciones del segundo tipo personaje, la cabra (figura 15), estuvieran finalizadas y listas para incluir en la primera prueba (aspecto no previsto inicialmente), por lo que fue posible que durante el primer experimento, los jugadores pudieran optar entre dos tipos de personajes.

El día 20 de enero organizamos una reunión con algunos compañeros que se ofrecieron voluntariamente a formar parte de la prueba. La prueba consistiría en jugar cuatro partidas. En ellas, se aseguraría que se probarían todas las acciones posibles del juego, desde el control de menús al control de los personajes, analizando durante todo el tiempo las reacciones de los jugadores, observando el juego desde fuera (pues ninguno de nosotros dos formó parte de la prueba), recopilando información sobre fallos, detectando mecánicas que no cumplían su propósito o directamente no

resultaban estimulantes para los jugadores. Durante la primera partida, se pidió a los participantes que intentaran jugar de forma seria, tratando de lograr los objetivos que plantea el juego, es decir, tratar de ganar la partida. Una vez que comprobamos que no hubo ningún problema significativo, se dió libertad a los jugadores para que no se contuvieran y probaran cualquier acción que se les ocurriese, para así someter al juego a situaciones que no estaban específicamente tratadas o diseñadas durante el desarrollo. La prueba tuvo una duración aproximada de hora y media, contando tiempo de juego, así como tiempo entre partida y partida donde realizamos algunos cambios en parámetros del juego en función de los resultados de cada una de ellas.

En la siguiente sección del capítulo profundizamos en los resultados obtenidos durante ese primer experimento.

Figura 15: Captura de una de las animaciones del personaje Cabra

4.2.1 Experimento 1

Fueron las mecánicas que no resultaban estimulantes para los jugadores las que provocaron que las conclusiones que obtuvimos del primer experimento no fueran excesivamente positivas. Por un lado, el juego tenía el carisma y la originalidad que buscábamos desde el principio, lo que supuso un gran orgullo, pero ciertas mecánicas que diseñamos intentando aportar dinamismo y cooperación a la partida, resultaron poco útiles y confusas. A continuación, se analizan todas ellas, indicando por qué no daban el resultado esperado, y cómo actualizamos su diseño para darles el peso en la jugabilidad que pretendíamos inicialmente:

- Intercambio de gnomo entre dos jugadores del mismo equipo

Como ya se mencionó anteriormente, el personaje portador del gnomo es el foco principal de atención durante la partida. Los dos jugadores rivales sólo tienen como misión arrebatar el gnomo a ese jugador, y este se encuentra en una situación de vulnerabilidad constante. Es por tanto tarea del compañero de equipo del portador el ayudar a mantener el gnomo en posesión, para así lograr llevarlo hasta la zona de puntuación.

Pues bien, para ello tuvimos que diseñar mecánicas que le permitieran ayudar a lograr ese objetivo de forma cómoda y eficaz. La primera mecánica a su disposición es común a los rivales y funcionó a la perfección durante el experimento: los jugadores que no portan el gnomo son capaces de realizar placajes que derriban al jugador con el que impactan. Sin embargo, esta acción no es siempre útil. Por ejemplo, si los dos jugadores rivales se encuentran ya cerca del portador, es difícil que el compañero sea capaz de derribar a los dos rivales antes de que estos derriben al portador.

La solución a la que llegamos fue que, si el portador del gnomo fuese capaz de pasarlo a su compañero, provocaría un cambio en la forma de jugar. Volviendo al problema antes mencionado, si dos rivales se encuentran ya cerca del portador, pero esta vez el portador pasa el gnomo al compañero desmarcado, este pasará a ser portador pero encontrándose sin oposición para avanzar hasta la zona de puntuación. Así, los dos rivales tendrían que organizarse para evitar esta situación tan desventajosa, aportando estrategias y colaboración entre jugadores (objetivo que consideramos clave para un juego de estas características).

Para esta primera versión del videojuego, el sistema de pase funcionaba de la siguiente manera: el jugador portador del gnomo, al pulsar el botón del mando correspondiente, entraba en modo lanzamiento, donde, manualmente, podía indicar la dirección en la que deseaba realizar el lanzamiento.

Una vez lanzado, si el gnomo llegaba hasta la posición del compañero, este pasaba a ser el portador de este, y el pase era completado. Si, por el contrario, el gnomo topaba con un jugador rival, el gnomo impactaba contra él y este era derribado como si de un placaje se tratara.

Pensamos que sería interesante que el sistema de pase fuera totalmente dirigido por el jugador, premiando así la habilidad de dirigir el pase en pocos segundos antes de que los rivales se encontrasen demasiado cerca. Sin embargo, el resultado obtenido no cumplía dicho objetivo. La acción en la partida era demasiado frenética, provocando que el jugador portador no dispusiera de tiempo suficiente como para poder apuntar hacia su compañero y realizar el pase.

La situación resultante solía ser que el portador realizaba el pase pero lanzando el gnomo lejos de la posición del compañero, o directamente era derribado antes de que pudiera siquiera realizar la animación de lanzar. Debido a la dificultad de la mecánica, los jugadores terminaban prescindiendo de ella, limitándose el compañero del portador a realizar placajes. El componente táctico perdía excesivo peso y el juego pasaba a ser una sucesión de derribos sin sentido.

A pesar de que la mecánica era difícil de usar de forma efectiva, decidimos conservarla para aquellos jugadores que fueran los suficientemente habilidosos como para darle uso durante las partidas. Sin embargo, seguía siendo imperioso que hubiera una forma fácil de pasar el gnomo entre compañeros de equipo.

Barajamos varias ideas para crear una nueva mecánica de pases. La que finalmente optamos como la más correcta, fue crear un sistema de pases automáticos. El

portador del gnomo, simplemente pulsando un botón, se encararía automáticamente en dirección a su compañero de equipo realizando el pase (siempre y cuando se encontrase a una determinada distancia). Del mismo modo, el personaje receptor del pase, se desplazaría de manera automática para recibir el gnomo lanzado. Durante este pequeño lapso de tiempo, el personaje que recibiera el pase, quedaría fuera del control del jugador a los mandos, aunque al ser un tiempo tan reducido (centésimas de segundo), éste era casi inapreciable por el mismo. De esta forma se garantiza que el pase siempre se produce con éxito.

Fue una decisión acertada, pues al observar los resultados en el segundo experimento que realizamos (el cual se comentará en el siguiente punto), la vertiente estratégica y de juego en equipo que buscamos se vió gratamente mejorada.

- Sprint

Otra de las mecánicas que ya se encontraban implementadas en esta primera versión jugable de *Gnomore Gnomes,* permitía a los jugadores (que no eran portadores del gnomo) la habilidad de realizar una aceleración repentina que hacía al personaje recorrer un gran tramo de terreno de juego en mucho menos tiempo que desplazándose de forma normal.

El objetivo principal, cuando decidimos introducir esta mecánica en el jueg,o era la de conseguir que los jugadores pudiesen evitar ser derribados, o sorprender al portador del gnomo acercándose a éste mucho más rápido. Una vez más, no obtuvimos el resultado deseado. En su lugar los jugadores abusaban del uso de la mecánica pulsando indefinidamente el botón del mando correspondiente, desde el inicio hasta el fin de la partida. Llegamos a la conclusión de que esto ocurría, en gran parte, porque era una habilidad que otorgaba ventaja en el terreno de juego, pero, sin embargo, su uso no penalizaba al jugador de ninguna manera. Si ya era especialmente difícil mantener el gnomo en posesión, desplazarse más lento que el resto de jugadores hacía la tarea de llevar el gnomo a la zona de puntuación prácticamente imposible.

La idea de que los jugadores pudieran realizar *sprints* durante el juego nos parecía atractiva, ya que aporta una capa de profundidad a la jugabilidad. Sin embargo, después de hablar detenidamente sobre las impresiones que esta mecánica nos había producido durante las partidas de prueba, decidimos rediseñarla completamente. El efecto jugable que queríamos conseguir era que fuera una acción útil, pero que su uso descuidado supusiera una desventaja.

El nuevo diseño de la acción de *sprint* rechaza que se trate de un breve “acelerón”, coloquialmente hablando, en favor de un periodo de tiempo durante el cual la velocidad del personaje es superior. En el momento en el que un jugador pulsa el botón de *sprint* asignado, la velocidad del personaje al desplazarse es significativamente superior.

Su velocidad vuelve a su valor natural si bien el jugador deja de pulsar el botón, o un determinado tiempo es superado. Una vez llegado a ese punto, el tiempo de *sprint* permitido se va recuperando hasta llegar a un valor máximo. La velocidad a la que el

tiempo permitido se recupera no será la misma a la cual se consume cuando se está realizando el *sprint.* Esa es la penalización que buscábamos conseguir dentro de la ventaja que supone estar en modo *sprint:* abusar de él puede provocar que rápidamente el tiempo permitido se consuma, y mientras el jugador espera a que el tiempo se recupere (la velocidad de recuperación es más lenta), otro jugador que ha administrado mejor su tiempo de *sprint* le supere en velocidad en alguna situación clave donde llegar primero sea importante (por ejemplo para coger el gnomo que está liberado).

Una vez diseñada la mecánica, era importante encontrar el mejor modo de indicar visualmente a los jugadores cuánto tiempo de *sprint* tenían disponible en cada momento. La solución fue la de colocar en las esquinas superiores de la pantalla dos barras en cada una (cada barra perteneciente a un jugador,) que se iban llenando o vaciando en función del uso de la acción de sprint que realizaban. Si un personaje tenía todo el tiempo de *sprint* para consumir, la barra aparecía completamente llena. Si, por el contrario, habían consumido todo el tiempo disponible, esta aparecía vacía. Si el tiempo se estaba recuperando, la barra se llenaba en proporción. Como añadido, se programó un sistema que permitía asignar colores al porcentaje de barra deseado. En la versión que se presentó en el segundo experimento, la barra aparecía roja de 0% al 25% completado, amarilla hasta el 75%, y verde hasta el 100%.

El hecho que los jugadores debieran preocuparse por gestionar la velocidad del personaje para obtener ventaja en el terreno de juego, produjo un incremento considerable de la carga estratégica durante las partidas y así se consiguió paliar los problemas que la primera versión del *sprint* producía en la jugabilidad.

- Zona de puntuación móvil

Inicialmente en esta primera versión de *Gnomore Gnomes,* la posición de la zona en la cual el portador del gnomo había de entrar para lograr un punto para su equipo iba variando a los largo de la partida. En el momento en el que uno de los cuatros jugadores obtuviera el control del gnomo, la zona de puntuación aparecía en una parte aleatoria del el terreno de juego. El jugador había de portar el gnomo hasta ella en un tiempo determinado, pues la zona iba empequeñeciendo por momentos. Si el tiempo transcurría y la zona de puntuación desaparecía debido a su pequeño tamaño, el portador perdía automáticamente la posesión del gnomo, este se liberaba y tenía que ser cogido de nuevo. Si un nuevo jugador lo cogía, la zona de puntuación volvía a cambiar de posición, y así a lo largo de la duración de la partida.

En un principio optamos por realizarlo de esta manera con la intención de que la partida se desarrollará en todas las zonas del escenario y no sólo en ciertos puntos del mismo. El resultado conseguido con esta mecánica no era del todo malo, pero generaba confusión entre los jugadores. Al ser partidas tan rápidas y frenéticas, la posesión del gnomo pasaba de un equipo a otro constantemente, y el tiempo que tardaba el jugador portador en ver hacia qué dirección debía dirigirse con él, ya había sido derribado por un rival (debido también a los dos problemas que se acaban de analizar).

La forma en la que esta mecánica afectaba a la partida no llegó a parecernos del todo incorrecta, pero no era suficientemente sólida (al menos no el modo de juego predeterminado). Así que finalmente optamos por emplear un sistema de zonas de puntuación fija (una para cada equipo), con una función similar a una portería en fútbol, o una canasta en baloncesto. Sin embargo, buscamos la forma de aportar alguna novedad que hiciese original la manera de jugar y diferenciara a *Gnomore Gnomes* con respecto a otros videojuegos de carácter deportivo.

El sistema resultante consistió en una zona de forma rectangular marcada sobre el terreno para cada uno de los equipos. Sin embargo, queriendo dotar de profundidad a la acción de puntuar, no queríamos limitarnos a que el jugador portador llegara con el gnomo al límite de la zona de puntuación (perteneciente al equipo contrario). Queríamos que la acción de puntuar fuera la recompensa por una serie de acciones consecutivas, que culminaran con llevar el gnomo a la zona rival. Por ello, se definieron tres acciones que permitían acumular méritos para puntuar:

- Completar un pase con éxito
- Proteger al compañero portador derribando a alguno de los dos rivales
- Golpear al gnomo (acción sólo realizable por el jugador portador)

Para mostrar el progreso de realizar dichas acciones, se añadió a la interfaz de usuario durante la partida una nueva barra (esta vez una por equipo), que muestra cuántas acciones más hacen falta para optar a puntuar, pues cada vez que una acción de la anterior lista era completada, la barra se completaba un porcentaje determinado. En el momento en el que la barra esté completa, la zona de puntuación del equipo rival era habilitada para puntuar. Se añadieron detrás de la zona cuatro modelos en tres dimensiones de antorchas que ardían en llamas mientras la zona estuviera habilitada. Si el equipo logra puntuar, o si pierde la posesión del gnomo, su barra pasa a estar vacía.

La idea inicial respecto a las porterías quedó descartada para el modo de juego básico, aunque no descartamos en un futuro el incluirla de manera opcional, como un modo de juego alternativo seleccionable por los jugadores.

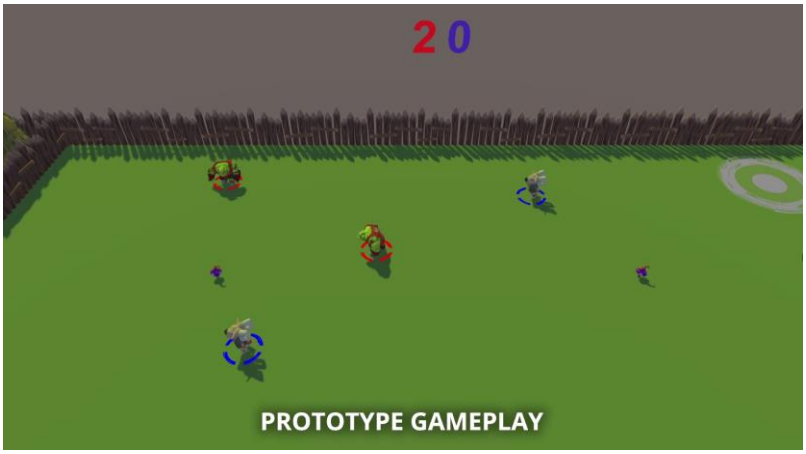

Figura 16: Captura de una de las partidas que tuvo lugar durante el primer experimento

Cuando concluyó el primer experimento, la planificación se adaptó para corregir todos los aspectos anteriores, por lo que algunas tareas que aparecían como ya terminadas, volvieron a la etapa de diseño, como eran las reglas básicas de la partida o la funcionalidad del personaje. Las tareas de rediseño, programación y pruebas de la funcionalidad corregida se alargaron hasta la primera semana de Marzo. Fue un proceso agotador y anímicamente más duro por tratarse de rehacer tareas en las que se había invertido ya mucho tiempo. Sin embargo, planificamos un sprint para tener las mejoras completadas el 20 de Marzo, y así fue. Una vez llegada la fecha en la que se cumplía el plazo, decidimos parar el desarrollo dos semanas, a modo de vacaciones, para recuperar fuerzas y motivación. Fue por aquellas fechas cuando oímos hablar del evento que se detalla a continuación, y que constituyó el segundo experimento.

4.2.2 Experimento 2

Una de las metas que establecimos en la planificación del proyecto fue el tener lista una versión jugable, con las mejoras introducidas después del primer experimento, para poder presentarla en alguno de los eventos de testeo de videojuegos que se producen a lo largo del año. En estos eventos se ofrece al público la posibilidad de probar videojuegos aún en fase de desarrollo, con el objetivo de obtener feedback y como posibilidad de prueba real con público real. La asociación Estudiantes de Videojuegos de Valencia (AEV) anunció un evento de estas características en la cervecería *Meltdown*, en la ciudad de Valencia. El evento se produjo el 22 de abril de 2017.

Así pues, decidimos inscribirnos para someter nuestro prototipo al juicio de público real y observar el nivel de interés suscitado por el juego. Consideramos que la forma más directa de recoger opiniones y sugerencias era a través de una corta encuesta que concentrara las opiniones que más nos interesaba recibir. Durante el evento, cuando un grupo de jugadores terminaba la partida, se les pedía si podían rellenar la corta encuesta a través de Google Forms (llevamos al evento un dispositivo iPad con la encuesta cargada para facilitar la tarea). Todos los grupos tuvieron una respuesta positiva y realizaron de buen grado la encuesta. Cabe destacar que todos aportaron sugerencias en la última parte de la encuesta. En general, se trataban de jóvenes con una edad comprendida aproximadamente entre los 20 y los 30 años. De los once encuestados, nueve eran varones y dos, mujeres. Debido a la naturaleza del evento, los encuestados eran usuarios habituales de videojuegos. Algunos de ellos eran además desarrolladores que estaban presentando sus videojuegos durante el evento.

A continuación vamos a mostrar y comentar los resultados obtenidos y las conclusiones que obtuvimos:

1. ¿Qué escenario te ha gustado más?

La versión que presentamos en el evento Beers & Testing permitía jugar en dos escenarios. Ambos tenían características jugables únicas y diferentes entre sí. El primer escenario, llamado Mushboom Forest, incluye un sistema en el que aparecen objetos en el terreno de juego, representados como setas, que pueblan de manera

aleatoria el escenario y que son activados por los jugadores al hacer contacto directo. Las setas otorgan una ventaja estratégica a los jugadores que las utilizan, como ralentizar a los enemigos cercanos, o derribarlos. Por otro lado, en el segundo escenario, llamado Rock'n Troll Hills, se generaban dinámicamente rocas gigantes que descendían por las colinas del escenario y atravesaban la zona de juego obligando a los jugadores a esquivarlas para no ser aplastados por ellas. En este último escenario, más que aportar nuevas estrategias de juego, se premiaba la habilidad de los jugadores, que debían estar atentos a más variables. Como podemos observar, la votación obtenida para cada uno de los escenarios fue muy similar, obteniendo una aceptación algo mayor para el escenario Mushboom Forest. Debido a ello, para siguientes escenarios deberíamos tener más en cuenta que resultan más atractivas las mecánicas que aportan componentes estratégicos a las partidas.

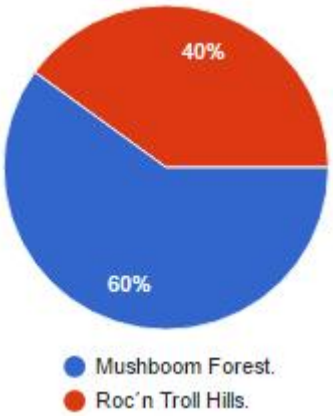

Figura 17: Resultados de la pregunta 1

2. ¿Te ha costado entender las mecánicas del juego?

Una de las mayores preocupaciones que teníamos antes de presentar el juego ante público real era que la curva de aprendizaje de nuestro juego fuese demasiado elevada (sobre todo en este tipo de eventos, donde el tiempo que el asistente dedica a cada juego suele ser mucho menor que el de un comprador) y resultase en una experiencia frustrante. Sin embargo, gracias a los cambios realizados a partir del primer experimento, la curva de aprendizaje no fue un problema, como podemos observar en el diagrama. La gran mayoría de los jugadores indicaron entendieron perfectamente las mecánicas y las reglas del juego. Sólo un pequeño porcentaje de los jugadores indicaron que les había costado un poco terminar de comprender el objetivo. Tenemos en cuenta estos resultados y está planeado el introducir un tutorial para mostrar los conceptos básicos del juego. Adicionalmente, otra de las ideas que manejamos para mejorar la asimilación de las reglas de la partida es mostrar imágenes explicativas durante las pantallas de carga.

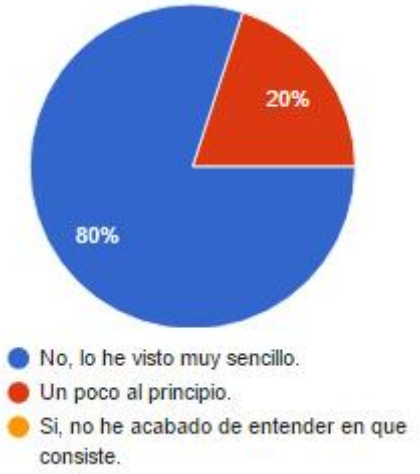

Figura 18: Resultados de la pregunta 2

3. ¿Qué valoración le darías al arte y las animaciones del juego?

El arte también fue uno de los grandes retos a los que nos enfrentamos a la hora de crear el videojuego. El diseño de personajes y escenarios es muy engañoso, pues un personaje o escenario que a vistas del desarrollador pueda parecer original y vistoso, puede no generar esa reacción en el público. Por lo tanto era de nuestro interés ver hasta qué punto nuestro esfuerzo era aceptado por jugadores. Aún siendo un prototipo, no una versión definitiva, gran parte de los elementos estéticos que se mostraban (modelos, animaciones, partículas) estaban prácticamente finalizados. La valoración del arte tuvo una aceptación bastante buena, obteniendo una única respuesta donde un usuario indicó que el apartado visual del videojuego no le había llamado especialmente la atención.

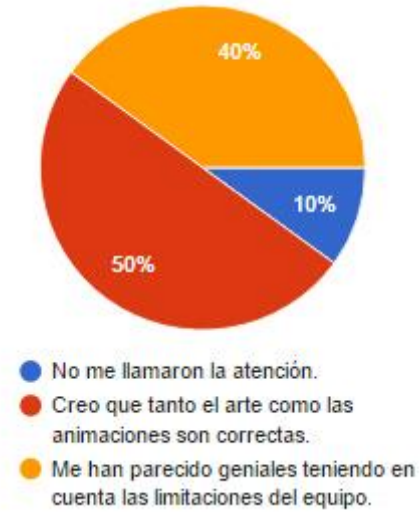

Figura 19: Resultado de la pregunta 3

4. ¿Te has sentido útil para tu equipo cuando tu compañero llevaba el Gnomo?

Uno de los problemas más latentes durante el primer experimento que realizamos en enero, fue que toda la atención de la acción que sucedía en el juego durante la partida se centraba únicamente en el jugador portador del Gnomo. Las mejoras realizadas tuvieron como cometido solucionar este problema y, como podemos observar en el gráfico, todos los encuestados afirmaron que cuando no poseían el gnomo colaboraban de distintas formas a favor de su equipo para conseguir la victoria, por lo que fueron bastante acertadas. El resultado de esta pregunta de la encuesta fue la más satisfactoria, ya que era uno de los problemas más graves que poseía la jugabilidad y en el que más tiempo invertimos para poder solucionarlo, rediseñando, implementando y probando una y otra vez las mecánicas básicas.

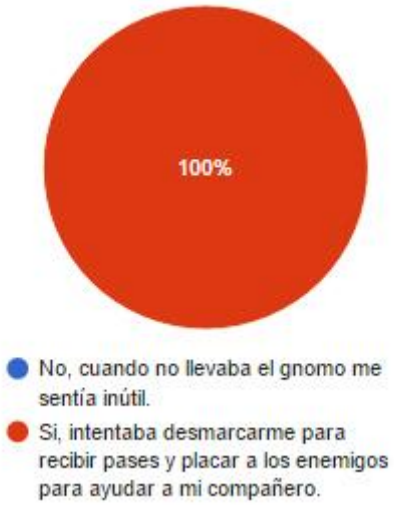

Figura 20: Resultado de la pregunta 4

5. ¿Crees 10€ por este juego sería un precio razonable cuando esté terminado e introduzca más escenarios, personajes, habilidades especiales y futuras expansiones?

Con esta pregunta pretendíamos averiguar si el modelo de negocio que habíamos elegido era factible o si por el contrario los usuarios consideraban que era un juego que debía ser gratuito. Del mismo modo también buscábamos acotar el precio de venta valorando las distintas opiniones por parte de los asistentes. Como se observa en las respuestas, un 80% del total de los encuestados opinaron que 10€ sería un precio razonable para el producto final y el resto aceptaría el hecho de que fuese algo más caro del precio que inicialmente teníamos pensado.

Figura 21: Resultado de la pregunta 5

6. ¿Qué valoración le darías a Gnomore Gnomes?

En la pregunta que concluía la encuesta, pedimos a los usuarios que valoraran el juego en una escala del 1 al 10 (siendo 1 la menor puntuación y 10 la mayor posible). Los resultados obtenidos fueron muy positivos, concentrándose todas las valoraciones entre el 7 y el 8 (llegando a obtener incluso una valoración de 10). Fue un buen resultado teniendo en cuenta que era un prototipo no exento de fallos y sin toda la funcionalidad implementada.

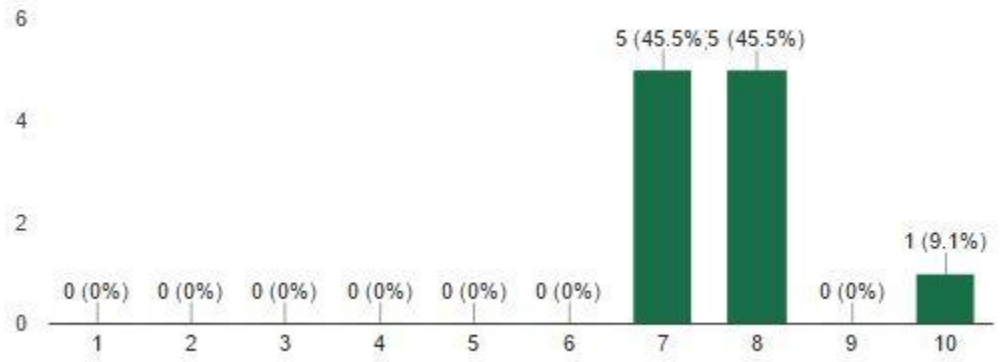

Figura 22: Resultado de la pregunta 6

7. Sugerencias u observaciones personales.

De forma opcional, dejamos un espacio al final de la encuesta donde incluir sugerencias u observaciones. Algunos de ellos se prestaron a dar su opinión y obtuvimos *feedback* valioso, con algunas propuestas muy interesantes sobre algunos elementos jugables que no nos habíamos planteado. Por ejemplo, varios usuarios insistieron en aspectos como diferenciar mejor los roles de los distintos personajes o equilibrarlos para que ninguno tuviese ventaja sobre el resto. Nos alegramos de ver que un usuario estaba interesado en el hecho de que el juego añadiera una mayor cantidad de personajes jugables y existiese la opción de poder personalizarlos, pues era una de las características que teníamos pensada incluir en el producto final. Otro usuario comentó que le había parecido confusa la elección de algunos botones del mando para ciertas acciones del personaje. Una de las sugerencias más interesantes fuen acerca de formas de mejorar algunos elementos de la interfaz de usuario (última observación en la figura 23).

Con el éxito durante el evento *Beers & Testing* finaliza la descripción del proceso que nos ha llevado a tener listo el mínimo producto viable. Desde este punto, hasta su puesta a la venta, están planificados tres experimentos más, siendo el tercero el más exhaustivo. En este, participaremos directamente en las pruebas, pues no serán partidas comunes las que se pondrán a prueba. En este tipo de pruebas, se realizan las diferentes acciones una y otra vez, buscando anomalías. Cada acción es posible que se haya de repetir varios cientos de veces, así como su combinación con el resto de acciones, por lo que este tipo de pruebas suele extenderse varios días.

A día de escribir esta memoria, nos encontramos trabajando en la presentación de la campaña de *crowdfunding* que nos permita financiar el resto del desarrollo del videojuego completo.

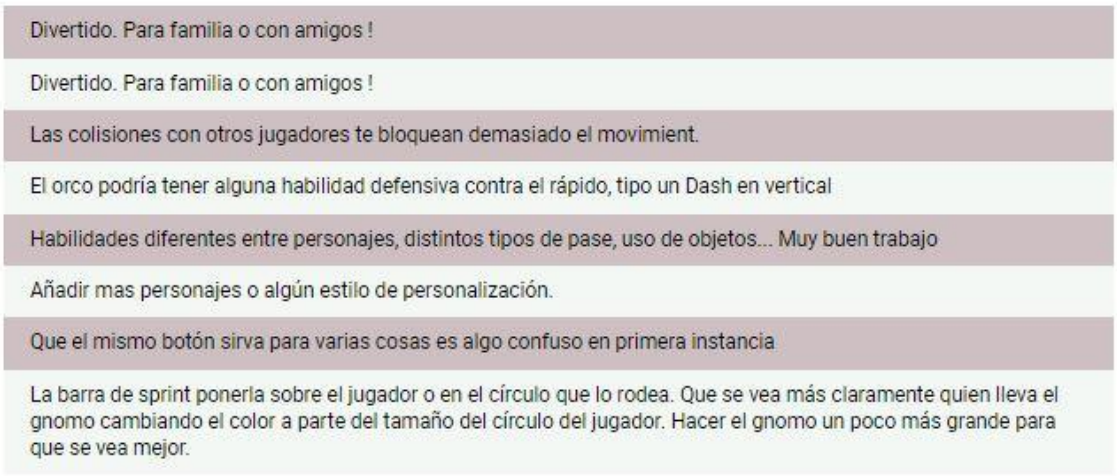

Figura 23: Resultado de la pregunta 7

5.Tecnologías utilizadas

5.1 Entorno de desarrollo: Unity

Antes de entrar en detalle sobre las características y funcionales que ofrece el motor gráfico Unity, vamos a definir qué es un motor gráfico. Un motor gráfico es un software que permite al desarrollador construir mediante sus herramientas un videojuego, ya que un motor gráfico aporta las herramientas necesarias para ello. Utilizando un símil con la construcción, podría decirse que un motor gráfico proporciona la maquinaria necesaria para construir el edificio. En este caso, el desarrollador tomaría el papel de jefe de obra, indicando qué hacer con cada máquina de construcción. Algunas de las más importantes 'máquinas' que un motor gráfico ofrece para la construcción de un videojuego son: motores de renderizado, detección de colisiones físicas entre objetos, animación, o audio. En la actualidad, hay un gran número de motores gráficos profesionales disponibles, siendo destacables los motores Unity, Unreal Engine, Cry Engine y Source Engine.

Al comenzar el desarrollo de un videojuego, la elección del motor gráfico es capital. Después de un estudio de los principales motores disponibles, la elección fue Unity. La elección fue debida a que es el motor más orientado al desarrollo independiente, con una curva de aprendizaje más corta que los demás motores, más orientado al desarrollo profesional para estudios experimentados. Otro motivo de peso es que se trata de un software gratuito. Si bien existe una versión Pro con funcionalidades adicionales, no eran necesarias para el desarrollo de nuestro proyecto. Otro aspecto importante de Unity es su facilidad para exportar a un amplio número de plataformas (más de 25, entre las que se encuentran plataformas del peso de PlayStation 4, Xbox One, iOS y Nintendo Switch). A pesar de que esa característica puede hacer mella en el rendimiento del videojuego, no suponía un problema para nuestro desarrollo, pues no es un software que demande un alto procesamiento gráfico.

Unity fue lanzado el 30 de mayo de 2005, y desde entonces no ha dejado de evolucionar. A fecha de esta memoria, la versión actual es la 5. Gracias a su flexibilidad y accesibilidad, se ha convertido en el motor gráfico más utilizado del mercado. Algunos de los videojuegos más destacados implementados en Unity son Ori And The Blind Forest, Hearthstone, Firewatch y Pokemon GO.

Figura 24: Algunas de las plataformas a las que Unity puede exportar

5.1.1 Interfaz gráfica de usuario

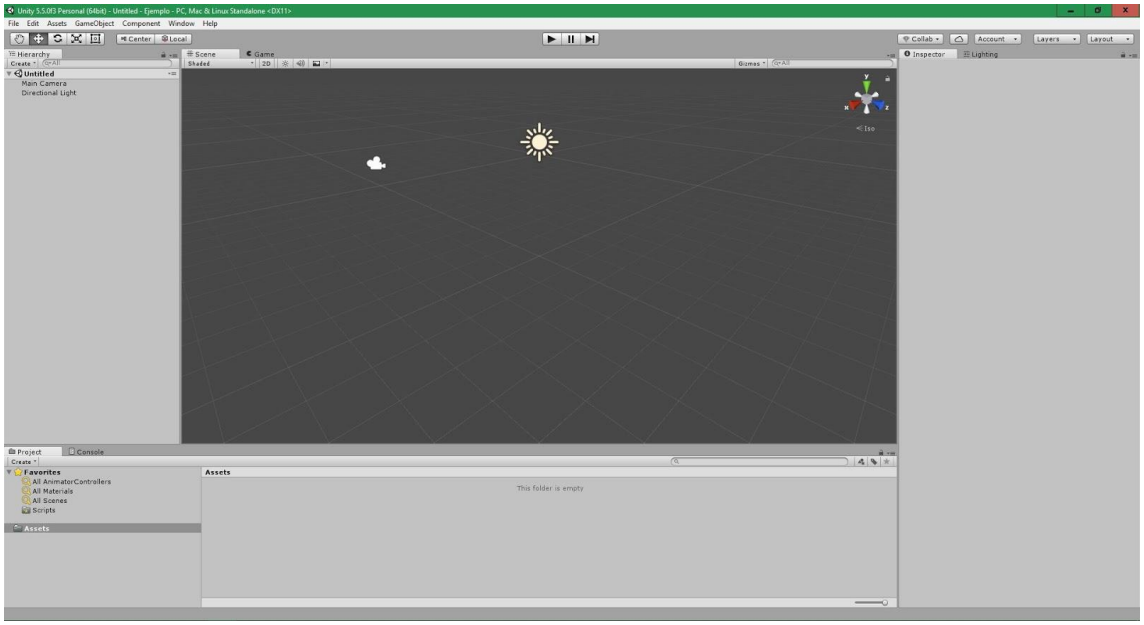

Figura 25: Vista principal de Unity al crear un proyecto nuevo

La interfaz gráfica de usuario en Unity destaca sobre las pertenecientes a otros motores gráficos por ser la más sencilla e intuitiva. La interfaz básica se compone de cuatro secciones, siendo este número variable, pues el usuario tiene completo control sobre como administrar y personalizar las distintas ventanas. Sin embargo, en esta memoria, nos centraremos en la configuración por defecto.

Como se observa en la figura anterior, y aunque algunas de las secciones se detallarán más exhaustivamente en capítulo siguiente, la interfaz básica se encuentra dividida en cuatro secciones: una central, dos laterales y una inferior.

La zona central, la más visible, tiene dos vistas que pueden alternarse mediante las pestañas que se encuentran en su parte superior: estas son la zona de trabajo, y el visualizador del juego. La zona de trabajo (*Scene View)* permite al desarrollador manipular los objetos de la escena, pudiendo así seleccionarlos, desplazarlos, escalarlos, etc. Podríamos decir que su función es la de construir la escena. La otra pestaña muestra el visualizador del juego (*Game View).* Esta vista muestra el juego resultante en cada momento (figura 26), destacando que si el desarrollador pulsa el botón *Play* que se encuentra en la parte superior de la interfaz, el juego se ejecuta y el desarrollador puede probarlo como si se hubiera exportado en un ejecutable.

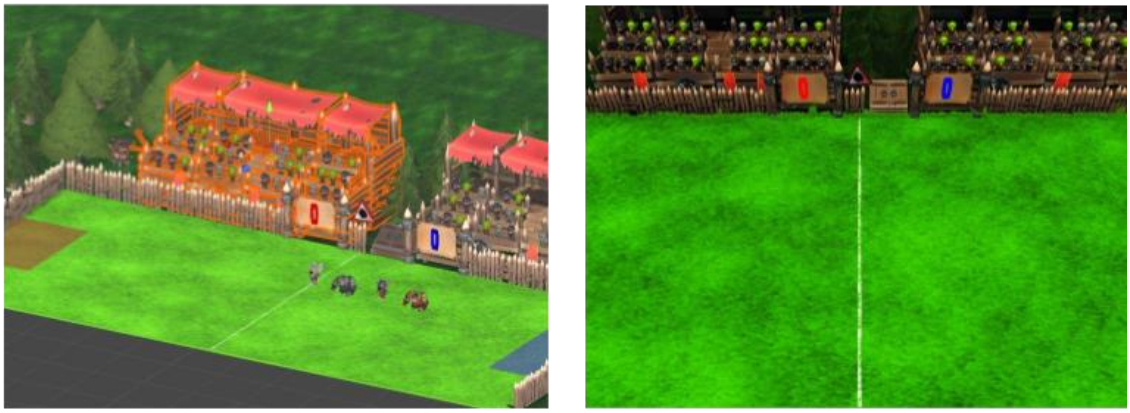

Figura 26: Comparación de la *Scene View* y de la *GameView* durante el desarrollo de uno de los escenarios de *Gnomore Gnomes*

Las dos secciones en los laterales se desarrollan con más profundidad en el siguiente capítulo, por lo que basta decir que la sección en el lateral izquierdo muestra la lista de objetos que se encuentran en la escena del videojuego en la que se está trabajando, y el derecho muestra los componentes del objeto que se encuentra seleccionado en la sección anterior, permitiendo así modificar sus valores.

Por último, la sección inferior cuenta con dos vistas, al igual que sucedía con la sección central. En la primera, llamada “Project”, el desarrollador tiene acceso a todos los ficheros que componen el proyecto: texturas, modelos, scripts, archivos de audio, etc. A estos ficheros se los conoce como “Assets” del proyecto. La segunda pestaña, “Console”, funciona como cualquier consola en un entorno de programación. La función de la consola es la de mostrar información para el desarrollador, como mensajes de error o de aviso.

5.2 GitHub

GitHub es una plataforma de desarrollo colaborativo de software, que facilita el desarrollo de proyectos en equipo permitiendo alojar proyectos en repositorios en línea. Su mayor atractivo es que amplía las posibilidades que ofrece el sistema Git, un sistema de control de versiones. Git permite a los desarrolladores mantenerse actualizados con cada modificación que alguno de los desarrolladores haga, así como crear ramificaciones, o volver a versiones más antiguas. Aparte de dichas características, GitHub ofrece herramientas para el desarrollo de una Wiki del proyecto, un sistema de seguimiento de problemas, una herramienta de revisión de código, y un visor de ramas que ofrece una perspectiva más visual del repositorio.

Durante el proyecto, el tener almacenado el proyecto en un repositorio Git fue fundamental para llevar un control de las diferentes versiones del prototipo. Así, si al implementar una nueva sección del videojuego algo que anteriormente funcionaba correctamente se estropeaba, bastaba con acceder al repositorio y retroceder a la última versión correcta almacenada.

Figura 27: Ejemplo de proyecto diversificado en varias ramas de un proyecto en Git

6. Desarrollo de la Inteligencia Artificial del personaje “gnomo”

Desde que planteamos el primer esbozo de lo que podría ser el videojuego Gnomore Gnomes, supimos que el elemento que más atracción suscitaría, que suponía el gran valor añadido del título, y sobre el cual giraría la experiencia, era el gnomo. Tanto en estética como en comportamiento, tenía que destacar sobre el resto de elementos del juego. El objetivo era lograr un personaje hilarante, que apareciera en pantalla corriendo despavorido por el terreno de juego, aterrado de los personajes controlados por los jugadores, dando vueltas corriendo mientras busca una salida o siendo lanzado por los aires de un extremo al otro del campo.

Para lograrlo, se precisaba de una inteligencia artificial que reaccionara a los movimientos de los cuatro jugadores, pero no de forma completamente predecible o perfecta, si no que debía simular inseguridad, llevando al gnomo a poder cometer errores en su tarea de huir.

Además del comportamiento indicado, había que dotar al gnomo de una acción inicial que acabara con este en el centro del terreno de juego, para dar lugar al comienzo de la partida una vez se encontrara en él. Tras valorar algunas posibilidades, se concluyó que el gnomo debía ser expulsado desde alguna plataforma (en los dos mapas incluidos en el prototipo se trataban de una puerta y de un géiser, respectivamente), que catapultara al pequeño gnomo hasta el centro del campo recorriendo una pronunciada parábola.

Antes de adentrarnos en el proceso de implementación de las dos acciones descritas, vamos a introducir la estructura del proyecto en el entorno de desarrollo Unity, para, a partir de ella poder explicar mejor cómo se introdujo el comportamiento del gnomo.

6.1 Estructura del proyecto en Unity

El motor gráfico Unity utiliza como elemento básico el GameObject. Un GameObject es usado para representar cada elemento que se encuentre en la escena del videojuego en cuestión: un objeto, un personaje, un controlador, el cielo, una pared invisible, un elemento de la interfaz gráfica de usario, etc.

A su vez, un GameObject está formado por componentes, que, como bien dice su nombre, componen las distintas partes de un GameObject. Por ejemplo, un GameObject usado para representar un personaje, precisará de un componente “Mesh Renderer” que incluya el modelo en tres dimensiones del mismo, un componente “Collider” (o caja de colisiones) que detecte las colisiones con los demás elementos de la escena, un componente “Animator” que gestione las animaciones del personaje, un componente “Script” que le de funcionaidad, y así tantos como el desarrollador crea conveniente.

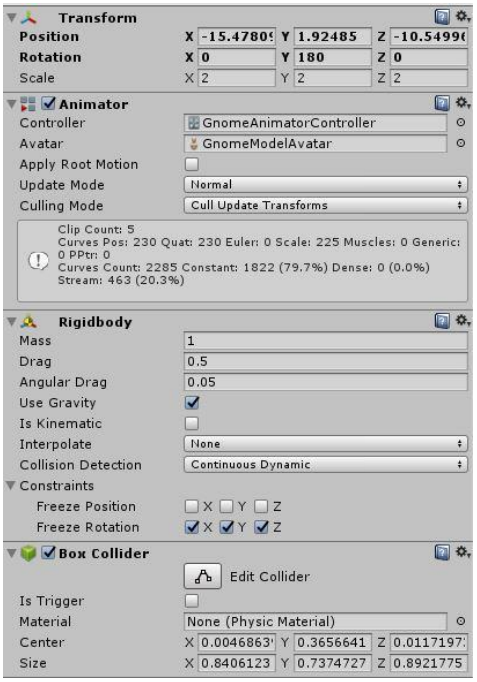

Figura 28: Conjunto de componentes de un determinado *GameObject*

La imagen que se encuentra a la izquierda es una captura de algunos de los componentes que forman el GameObject que representa al gnomo en la partida. En ella se pueden observar varios de los componentes más comunes en un GameObject. El primero, llamado “Transform”, es el componente común a todos los GameObjects. Este componente dota al elemento de una posición, una rotación y una escala en el mundo virtual. Por ejemplo, si incrementáramos el valor “X” que forma parte del campo “Position” perteneciente al componente, el gnomo se desplazaría hacia la derecha.

El segundo componente, “Animator”, controla las animaciones incluidas en el personaje, creando transiciones entre ellas según lo requiera el momento de la partida.

El tercer componente es uno de los más complejos, pero a la vez uno de los más importantes en Unity. El componente “RigidBody” se encarga de dotar al GameObject de comportamiento se vea controlado por la física. El experimento más sencillo a realizar para entender su funcionamiento es crear dos GameObject con forma de cubo (este tipo de GameObject se encuentra ya predefinido en Unity), uno en una posición más elevada que el segundo (manipulando el valor “Y” del campo position en el componente “Transform” como se ha explicado antes), y añadir un componente “RigidBody” al que se encuentre en la posición más elevada. Al ejecutar el juego, el cubo caerá afectado por las leyes de la gravedad, y al impactar contra la caja de colisión del otro GameObject (cuarto componente de la captura), rebotara de forma realista. Sin duda, se trata de un componente complejo que ha de ser manipulado con mucho cuidado. Más adelante en este capítulo se trata en mayor detalle este componente.

Pues bien, una vez definido la función de un GameObject, aún tenemos algunos aspectos más a comentar sobre Unity para que la explicación de los algoritmos que componen la inteligencia artificial del gnomo sean más comprensibles.

Un videojuego es un sistema reactivo, pues reacciona en función de las acciones del jugador, ejecutando el código correspondiente a cada situación que se de. Dicho código se encuentra en “Scripts”, ficheros de código escritos en el lenguaje de programación C#, y son un componente más a añadir a los GameObjects. Algunos scripts controlan los sucesos que están sucediendo en la partida, otros otorgan funcionalidad, otros organizan los distintos GameObjects activos, etc.

Figura 29: Jerarquía de los diversos *GameObjects* de la escena de juego

En la captura que aparece en la derecha (sección lateral de la interfaz de Unity introducida en el capítulo anterior), se puede ver la lista de GameObjects al inicio de la escena “Mushboom Forest” (uno de los dos escenarios presentes en el prototipo). A destacar por contener scripts importantes, se encuentra el “Scene Manager”, que contiene scripts que gestionan las acciones que van sucediendo en la escena, el “Main Camera”, que además de contener un componente “Camera” que permite visualizar la escena, contiene un script que mueve dicho componente para enfocar la parte del terreno de juego en la que se encuentra el gnomo, y el “Canvas”, que contiene los scripts necesarios para interactuar con la interfaz gráfica de usuario.

Este primer esbozo de cómo está estructurada la base de un proyecto en Unity nos sirve como punto de partida para explicar cómo se desarrollaron los scripts que dan vida al carismático gnomo durante el videojuego. Todos los aspectos de Unity que precisen de aclaración a lo largo del capítulo serán debidamente explicados durante el mismo.

6.2 Implementación de los algoritmos

6.2.1 Creación

Como se ha mencionado en el apartado anterior, el *GameObject "SceneManager"* contiene un script que gestiona los eventos que se suceden a lo largo de la escena. Concretamente, durante la escena de la partida, ha de crearse un gnomo al inicio de la escena, y cada vez que un jugador anote un punto. Por tanto, el script perteneciente a "SceneManager" debía hacerse cargo de iniciar la creación de cada uno de los gnomos, en el momento adecuado. A continuación analizamos los scripts (ficheros de código de extensión .cs) y las líneas de código más relevantes que intervienen en dicho proceso:

- **SceneBehaviour.cs**

Como se ha indicado anteriormente, inicialmente el gnomo es catapultado desde un elemento del escenario hacia el centro del terreno de juego. Centrándonos en el primer escenario, *Mushboom Forest*, este elemento se trata de un pequeño portón que se encuentra pegado al borde del campo. Este *GameObject* contiene un componente "Animator" que permite gestionar sus animaciones. En este caso, la animación que nos interesa es una en la que el portón se abre para dejar paso al gnomo.

Figura 30: Estado inicial y final de la animación que abre el portón

Pues bien, una vez seleccionada la animación que queremos manipular, aparece la ventana de Unity para gestionar las animaciones. En ella aparece desglosada la animación fotograma a fotograma, mostrando todos los cambios en el objeto en cada uno de ellos. Lo que nos interesa es buscar el fotograma en el que el portón se encuentra abierto, y colocar en el mismo un evento de animación. Un evento de animación es un sistema propio de Unity en el que, llegado al fotograma indicado por el desarrollador, activa un método del script indicado. En nuestro caso, el método es "GenerateGnome()", que, como bien dice su nombre genera un gnomo, perteneciente al script SceneBehaviour.cs. Las marcas en los distintos metrajes que se observan en la siguiente figura muestran los diversos eventos de animación establecidos en la animación del portón.

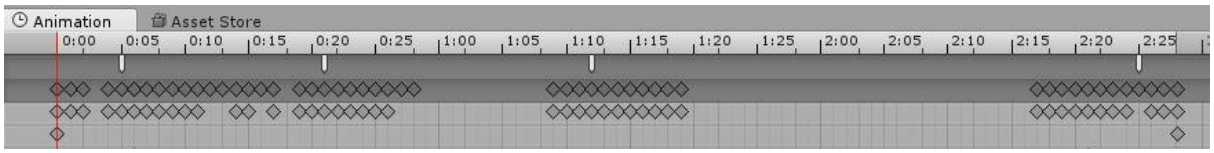

Figura 31: Línea de tiempo de una animación en *Unity*

Lo siguiente, pues, es definir el método *GenerateGnome().* Este método toma la posición en el mundo virtual del centro del portón, para instanciar en ese punto un *Prefab* del *GameObject* gnomo. Un *Prefab* es un *Gameobject* configurado previamente que se almacena como fichero en el proyecto, y así poder ser reutilizado a lo largo de él cómodamente. En este caso concreto, el *Prefab* "*GnomePF*", es un *GameObject* con los componentes que forman el personaje del gnomo.

A continuación mostramos las líneas de código más importantes del método *GenerateGnome()* que permiten generar un gnomo, con las explicaciones y anotaciones pertinentes:

Lo primero que hacemos en este método es obtener la posición (x, y, z) del punto de salida del gnomo. Para ello, en la variable declarada al principio del script _gnomeDoor, accedemos a la posición del GameObject que representa la puerta, y almacenamos su valor en la variable gnomePosition. Este tipo de variable (Vector3), es capaz de almacenar tres valores, usado normalmente para representar las tres coordenadas espaciales.

```
Vector3 gnomePosition=_gnomeDoor.transform.GetChild(0).transform.position;
```

La siguiente sección de código corresponde a la creación de un nuevo GameObject, que contendrá el nuevo gnomo. Esta variable, llamada gnomeGo, contiene el resultado de crear una instancia (una copia, para entendernos) del Prefab "GnomePF" en la posición que acabamos de almacenar, que se encuentra en la variable gnomePosition. Como detalle, cambiamos el nombre del nuevo GameObject que Unity establece por defecto por "Gnome", para que resulte más fácil encontrarlo en la lista de GameObjects activos en la escena.

```
GameObject gnomeGO = Instantiate(Resources.Load("GnomePF"),gnomePosition,
Quaternion.identity) as GameObject;
gnomeGO.name = "Gnome";
```

Mediante AddComponent, añadimos al GameObject gnomeGo el componente GnomeBehaviour, que es el script que gestiona el comportamiento del gnomo.

```
GnomeBehaviour gnomeBehaviour=gnomeGO.AddComponent<GnomeBehaviour>();
gnomeBehaviour.Initialize (this, _gnomeCount);
```

Por último, se accede al GameObject estático GnomesManager, que incluye un script que gestiona la información de los gnomos activos en escena (si está siendo portado por un jugador, que jugador fue el último en portarlo, etc.). Una vez accedido a la instancia del manager de gnomos, mediante el método NewGnome, se le comunica

que un nuevo gnomo ha sido creado, suministrándole el GameObject y la cuenta actual del número de gnomos creados.

```
GnomesManager.instance.NewGnome (gnomeGO, _gnomeCount++);
```

Esta es la parte del script *SceneBehaviour.cs* que interviene en la creación del gnomo. Ahora, nos centramos en el lanzamiento inicial del gnomo, controlado por el script *GnomeRelease.cs,* encargado de generar el lanzamiento del gnomo desde su punto inicial, hasta el punto central del terreno de juego.

6.2.2 Lanzamiento

- **GnomeRelease.cs**

Este script se activa al inicializar el script *gnomeBehaviour,* introducido anteriormente, pues este establece la acción "*Release*" como la acción inicial en el manejador de acciones, haciendo que el script asociado a la acción, *gnomeBehaviour,* fuera activado al principio de generarse el gnomo.El manejador de acciones es un sistema que desarrollamos como solución al problema de tener acciones sucediendo de forma simultánea en el personaje que llevaba a colisiones entre las mismas. Para ello, se desarrolló un algoritmo que permitía definir capas de acciones para así poder establecer eficazmente cuáles podían coexistir y cuáles no podían activarse si una contraria se encontraba activa.

Además el algoritmo organizaba de forma automática la gestión de animaciones para que dos acciones que estuvieran coexistiendo no se mezclaran dando resultados indeseados, ya que la gestión de animaciones de forma manual con un número elevado de situaciones puede resultar complejo y problemático, como se muestra en la figura a continuación.

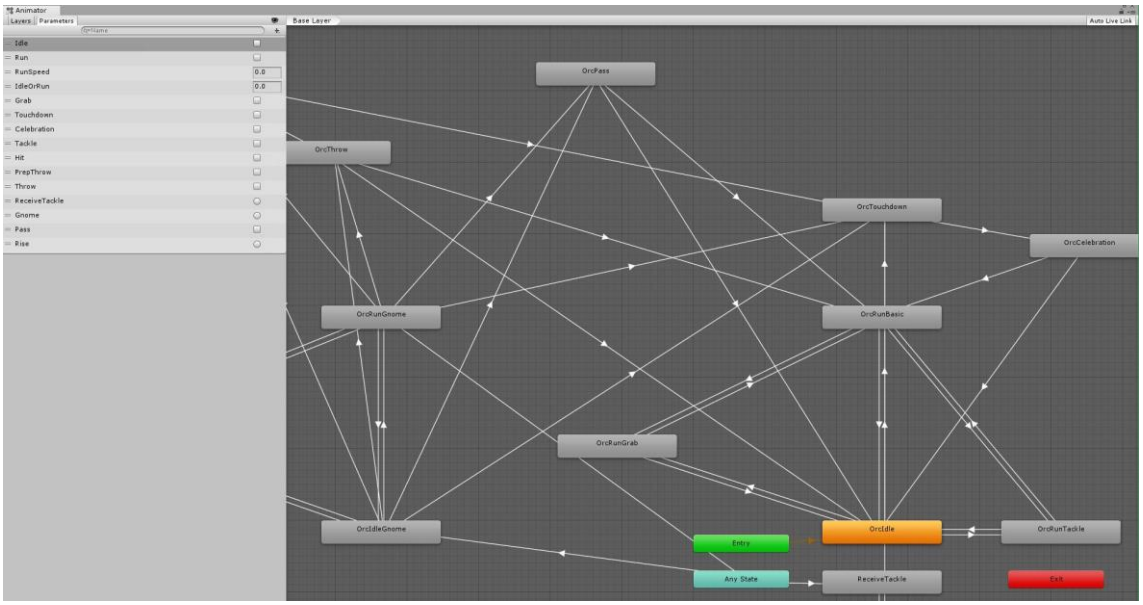

Figura 32: Captura del gestor de animaciones de *Unity* de uno de los personajes de *Gnomore Gnomes*

Este sistema fue diseñado con la ayuda de Francisco Muñoz Montoya, estudiante de Máster en la *Escuela Técnica Superior de Ingeniería Informática,* y aportó robustez a la jugabilidad del videojuego, que debido a la acción frenética y rápida de sus partidas, ha de gestionar sin fisuras la sucesión de acciones que se van realizando y terminando sin descanso.

Pues bien, el código que trata el lanzamiento del gnomo se divide en dos fases: cálculo inicial de la fuerza a la que el gnomo ha de ser lanzado para llegar al punto deseado, y la comprobación de que ha colisionado con el terreno de juego, finalizando así la acción "Release".

La primera parte del cálculo del lanzamiento calcula la velocidad vertical que habrá que aplicar al gnomo para alcanzar una altura determinada, en función del valor de la gravedad, y la velocidad horizontal que se habrá de aplicar para recorrer la distancia que separa al gnomo (transform.position), del centro del terreno de juego (distance). Las dos velocidades resultantes se almacenan en vSpeed y hSpeed, respectivamente.

```
float g = -Physics.gravity.y;
float vSpeed = Mathf.Sqrt(2 * g * 25);
float totalTime = 2 * vSpeed / g;
float distance = Vector3.Distance(transform.position, target);
float hSpeed = distance / totalTime;
```

La siguiente parte del algoritmo encara al gnomo hacia el centro del terreno de juego (almacenado en la variable target). Para ello se calcula el vector dirección y se actualiza la nueva rotación en el componente "Transform" del gnomo.

```
Vector3 lookPos = target - transform.position;
lookPos.y = 0;
Quaternion rotation = Quaternion.LookRotation(lookPos);
transform.rotation = rotation;
```

Una vez calculados las velocidades y la rotación, se procede a aplicarlas en el componente "Rigidbody", introducido al comienzo del capítulo. Más concretamente, se modifica el campo velocity del mismo para así aplicarle una fuerza al objeto (en este caso el gnomo). Inicializando los valores de velocity y angularVelocity a 0 para que no intervengan velocidades anteriores, aplicamos la hSpeed y vSpeed calculadas anteriormente para dar el impulso necesario para que este llegue al punto central del terreno de juego.

```
_rigidbody.velocity = Vector3.zero;
_rigidbody.angularVelocity = Vector3.zero;
_rigidbody.velocity = new Vector3(transform.forward.x * hSpeed,
vSpeed, transform.forward.z * hSpeed);
```

La segunda fase comprende la comprobación de que el gnomo ha colisionado con el suelo del terreno de juego. Para ello, necesitamos dos elementos propios de Unity. El

primero son las llamadas *Layers,* que permiten clasificar los *GameObjects* en categorías. Esto nos ayuda a la hora de detectar colisiones, pues podemos indicar que sólo detecte las colisiones con objetos pertenecientes a una determinada *Layer.* En nuestro caso, queremos que el gnomo detecte si ha colisionado con un *GameObject* perteneciente a la *Layer Scenario.*

El siguiente elemento que necesitamos se denomina *RayCast.* El *RayCast,* de forma simplificada, lanza un rayo virtual (en forma de línea recta) en una dirección y distancia establecida, y notifica si ha colisionado con un objeto perteneciente a una determinada *Layer,* como se muestra en la figura siguiente.

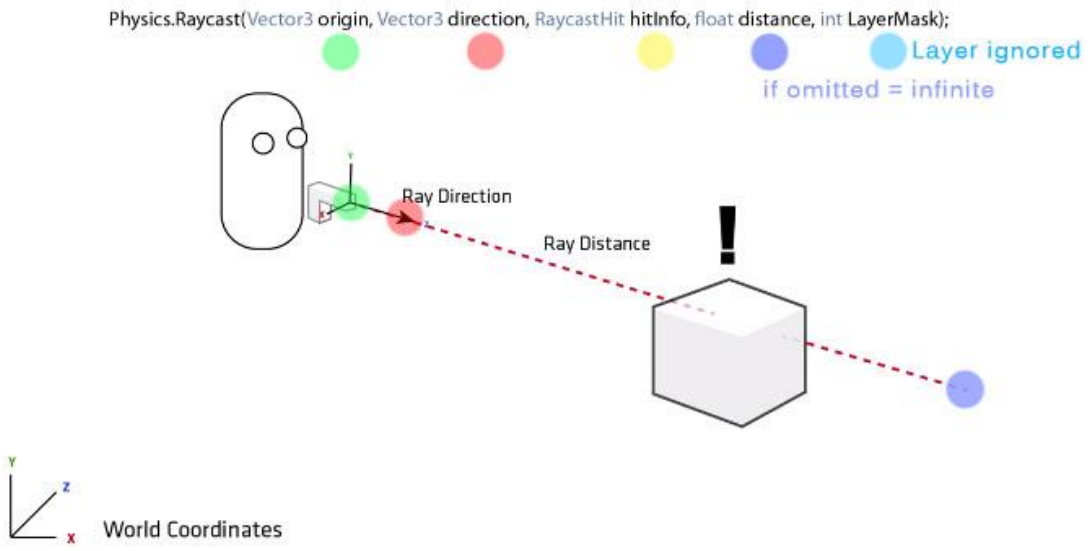

Figura 33: Ejemplo gráfico del funcionamiento de la función Raycast de Unity

Para nuestra situación, la línea tiene como origen la posición del gnomo, dirección (0, -1, 0) para que compruebe colisiones con objetos situados por debajo de la posición del gnomo, y longitud 0.1, para que solo detecte colisiones con objetos muy cercanos a la posición del gnomo. Esta comprobación se realiza una vez por fotograma. A continuación se muestra el segmento de código que logra dicha tarea:

```
int scenarioLayerMask = 1 << LayerMask.NameToLayer ("Scenario");
      Vector3 start = transform.position;

if (Physics.Raycast (start, Vector3.down, 0.1f, scenarioLayerMask)) {
      EndRelease ();
}
```

Si la condición indicada se cumple y, efectivamente, el gnomo se encuentra en colisión con el terreno de juego, se ejecuta el método EndRelease(), que notifica al script manejador de acciones que la acción “Release” ha finalizado. En ese momento, el manejador comprueba las condiciones que resuelven cuál debe ser la siguiente acción

a activar. Cuando la acción “Release” concluye, la acción que se activa es “Run”, que permite al gnomo moverse por el terreno de juego tratando de huir de los personajes. Es esta acción la que se detalla en la siguiente sección de este capítulo.

6.2.3 Movimiento

Esta es la acción más compleja de las que forman la funcionalidad del gnomo. Debido a la longitud y la complejidad del código que desplaza al gnomo por el terreno de juego, para esta memoria se va a tratar de describir cuál es el proceso mediante el cual el gnomo decide a qué dirección desplazarse en cada momento sin tener que introducir el código específico. A lo largo de la explicación se mostrarán imágenes del desarrollo del prototipo del sistema, con una abstracción de los escenarios y jugadores utilizando figuras geométricas simples. En dicho prototipo, los jugadores se representaban mediante cápsulas de colores rojo y azul, los límites del terreno de juego como paredes invisibles, y el gnomo como un cubo blanco, tal y como se muestra en la figura a continuación.

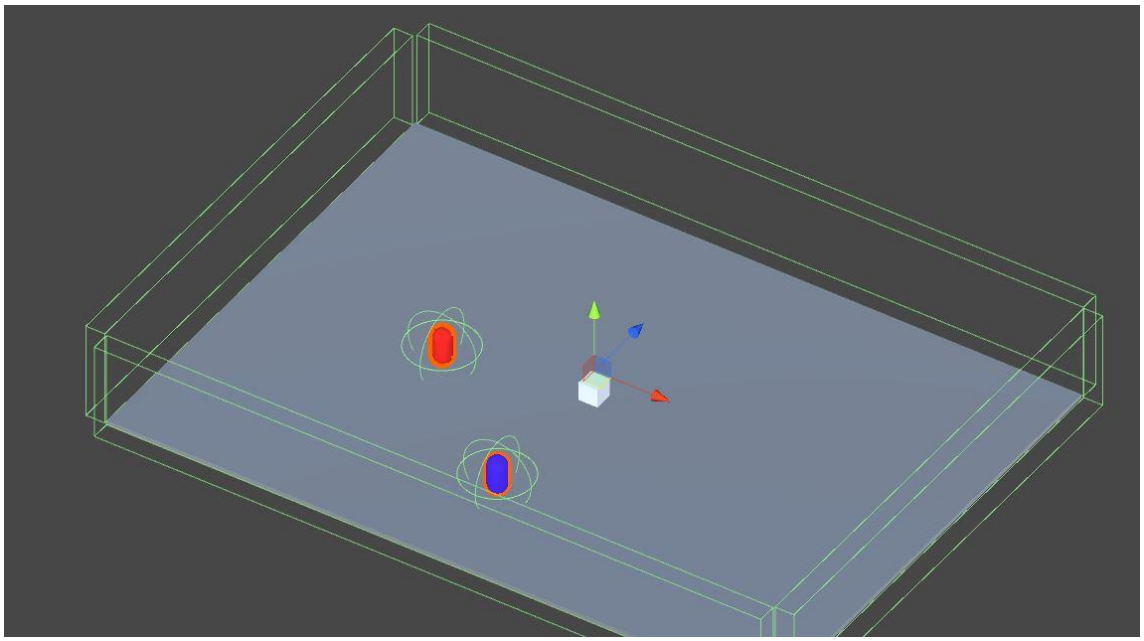

Figura 34: Espacios de colisiones de los límites del terreno de juego y de los diversos jugadores

La forma de proceder del gnomo es la siguiente: primero realiza un reconocimiento donde analiza la posición de cada uno de los cuatro jugadores respecto a su posición, así como la distancia a los muros que delimitan el terreno de juego. El reconocimiento se realiza lanzando rayos virtuales (*RayCast)* como en la comprobación del lanzamiento vista en la anterior sección, salvo que esta vez se lanzan de forma perpendicular al gnomo. Cada *RayCast* incrementa el ángulo del anterior, consiguiendo lanzar rayos de detección de colisiones alrededor del gnomo. El número de rayos puede modificarse, pudiendo perder así precisión, pero ganando rendimiento en caso de reducirse, pues el número de cálculos variaría. En la siguiente figura se muestran los rayos que lanza el gnomo para la detección de elementos en el terreno

de juego en forma de líneas verdes. En este caso concreto, el gnomo estaba configurado para lanzar 360 rayos cada vez.

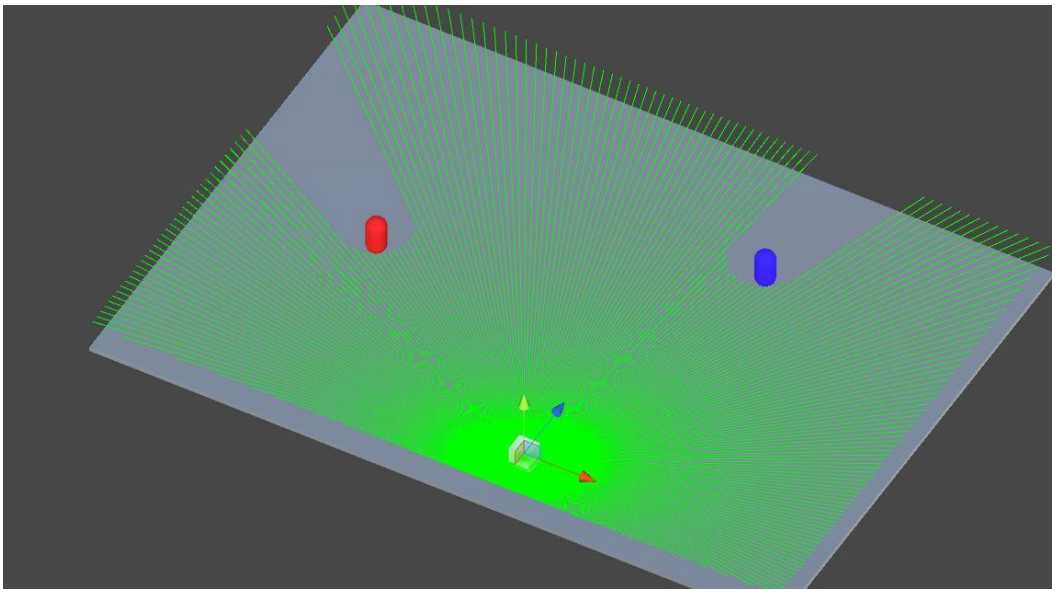

Figura 35: Lanzamiento de rayos de detección de colisiones desde el gnomo hacia todas direcciones

Conforme se van detectando las colisiones con los jugadores y con los límites del campo, se va calculando la dirección en la que más terreno de juego queda más despejado, pues es la dirección a la que más conviene que el gnomo se dirija. Esto se consigue normalizando la acumulación de los vectores dirección con respecto a los personajes y a los límites del campo. En la siguiente figura, se muestra una línea roja que indica la dirección elegida por el gnomo (representado por un cuadrado blanco) para huir.

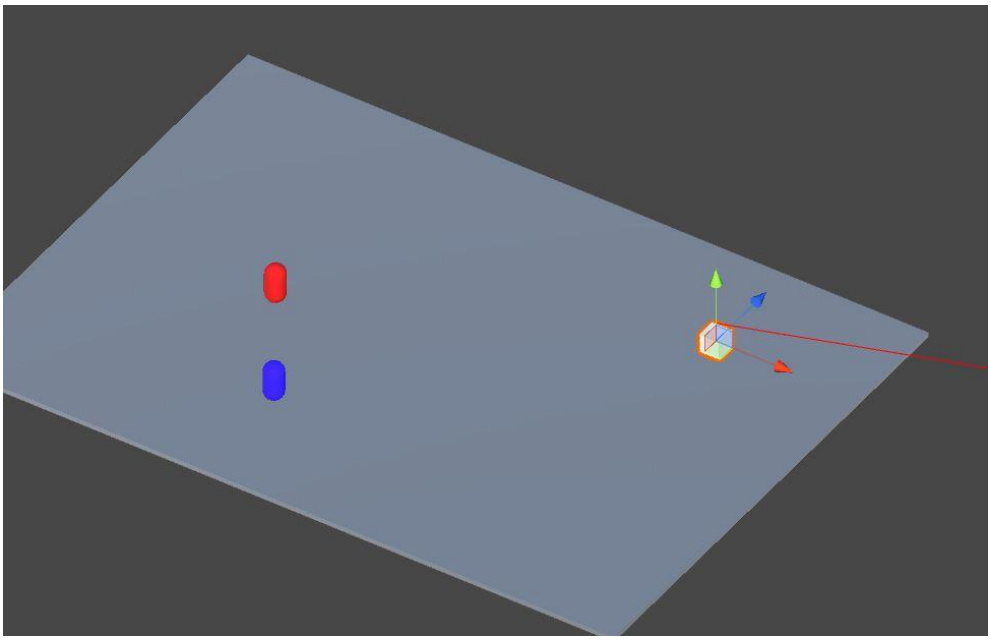

Figura 36: Dirección elegida por el gnomo para huir en un momento determinado con dos jugadores

Sin embargo, a pesar de que así se consigue la dirección más óptima a la que huir, huir hacia una zona donde no hay jugadores pero sí límite de campo es preferible que dirigirse hacia una zona en la que hay jugadores. Pero, tal y como se encontraba el algoritmo hasta ese momento, el gnomo huía tanto de los jugadores como del muro límite, limitando así mucho el rango de opciones por las que optar.

Por tanto, el detectar un jugador debería tener más peso en la decisión que el detectar un límite del terreno. Por ello, los vectores dirección acumulados están multiplicados por un peso que depende de si se trata de una dirección hacia la que se encuentra un jugador (mayor peso), o si se trata de la dirección hacia uno de los límites del escenario (menor peso). Con esto sí logramos el resultado deseado. Además, el algoritmo se adapta a cualquier número de jugadores, como se puede comprobar en la siguiente imagen, en la que se probó el sistema con 23 objectos actuando como jugadores. Las opciones para el gnomo en este caso son mucho más limitadas, como se puede apreciar en la dirección elegida (línea roja de la figura 37).

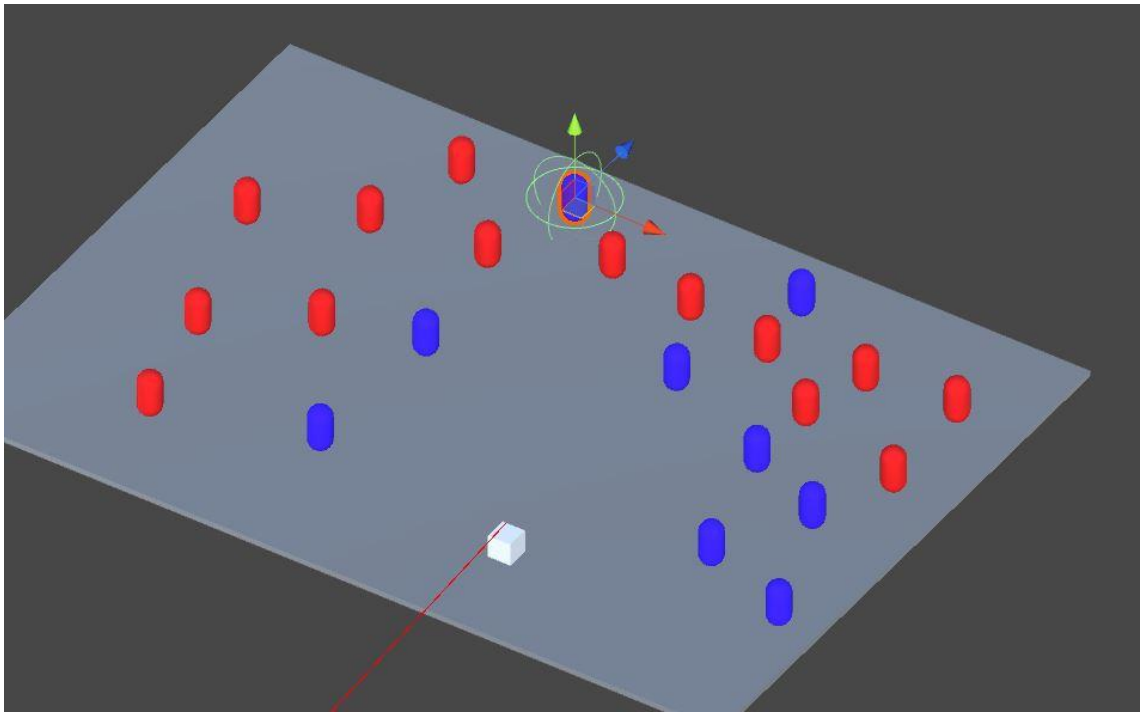

Figura 37: Dirección elegida por el gnomo para huir en un momento determinado con veintitrés jugadores

Aún así, era un sistema mejorable, pues queríamos dotar al gnomo de un comportamiento que le hiciera parecer más inseguro de sus decisiones, debido al pánico al que debería estar sometido si su situación fuera real. Para lograr una aproximación a ello, se aleatorizaron algunos componentes del algoritmo (dentro de unos rangos), para que diera la sensación de que toma decisiones precipitadamente, o de que incluso tomaba decisiones equivocadas algunas veces.

Para lograr simular su incertidumbre, el gnomo se replantea sus decisiones en intervalos de tiempo determinados por un valor aleatorio. De esa forma, en momentos

concretos, varía dos o tres veces de dirección en un intervalo de tiempo corto, dando la sensación de que está descartando opciones mientras se está moviendo, aportando mucho carisma al personaje. La acotación del intervalo sobre el que es elegido el valor aleatorio utilizado permite personalizar su comportamiento y así probar la sensación visual producida con distintos valores. En la versión actual del videojuego, el gnomo se replantea la dirección a tomar en intervalos de entre 0,5 y 1 segundos.

Para que no siempre elija la mejor opción hacia donde huir, tambíen se aleatoriza el rango de direcciones sobre las que puede elegir. Es decir, una vez calcula la dirección óptima de huida, se define, a partir de un ángulo establecido, un cono de visión para el gnomo. Así, al aleatorizar una dirección comprendida dentro de ese cono de visión, su elección no es siempre la más correcta, provocando desviaciones. Ampliar o reducir ese ángulo de visión provoca que el gnomo tome mejores o peores decisiones, por lo que se realizaron pruebas hasta encontrar el ángulo que mejor representaba el efecto que buscábamos sin comprometer la jugabilidad, pues seguía siendo necesario que el gnomo fuera lo suficientemente efectivo huyendo de los jugadores.

El resultado final fue gratamente satisfactorio, y fue uno de los aspectos más comentados de forma positiva por parte de los early-adopters que particparon en las diversas pruebas.

7. Conclusiones y trabajo futuro

Como se menciona al comienzo del cuarto capítulo, desarrollar un videojuego es una ardua tarea, pero, sobre todo, llena de incertidumbre. Hemos puesto toda nuestra pasión por los videojuegos en desarrollar *Gnomore Gnomes,* aunque eso nunca es garantía de éxito. Lo que a nosotros nos pueda parecer un videojuego lleno de carisma y diversión, al público puede parecerle mediocre y aburrido. Las primeras pruebas, que mostraban fallos de diseño y de programación, fueron descorazonadoras, y comenzamos a plantearnos si el videojuego tenía futuro.

Pero fue esa misma pasión por nuestro proyecto y por el mundo de los videojuegos la que hizo que no nos rindiésemos, y continuásemos batallando contra los obstáculos que nos separaban de nuestro objetivo. El objetivo de hacer vibrar y reír a los jugadores, como tanto habíamos vibrado y reído en nuestros videojuegos preferidos. Fue durante el evento *Beers & Testing X* donde, para nuestra sorpresa, provocamos esos sentimientos en todos los jugadores que se sentaron frente a la pantalla, cogieron los mandos, y jugaron a *Gnomore Gnomes.* No ha habido un momento de mayor orgullo y de mayor recompensa personal que ver a cuatro desconocidos levantarse de sus sillas de pura emoción disfrutando de un videojuego creado desde una hoja en blanco que empezamos a rellenar durante una de las muchas clases en la facultad que pasamos juntos. Hoja que se muestra en la siguiente figura:

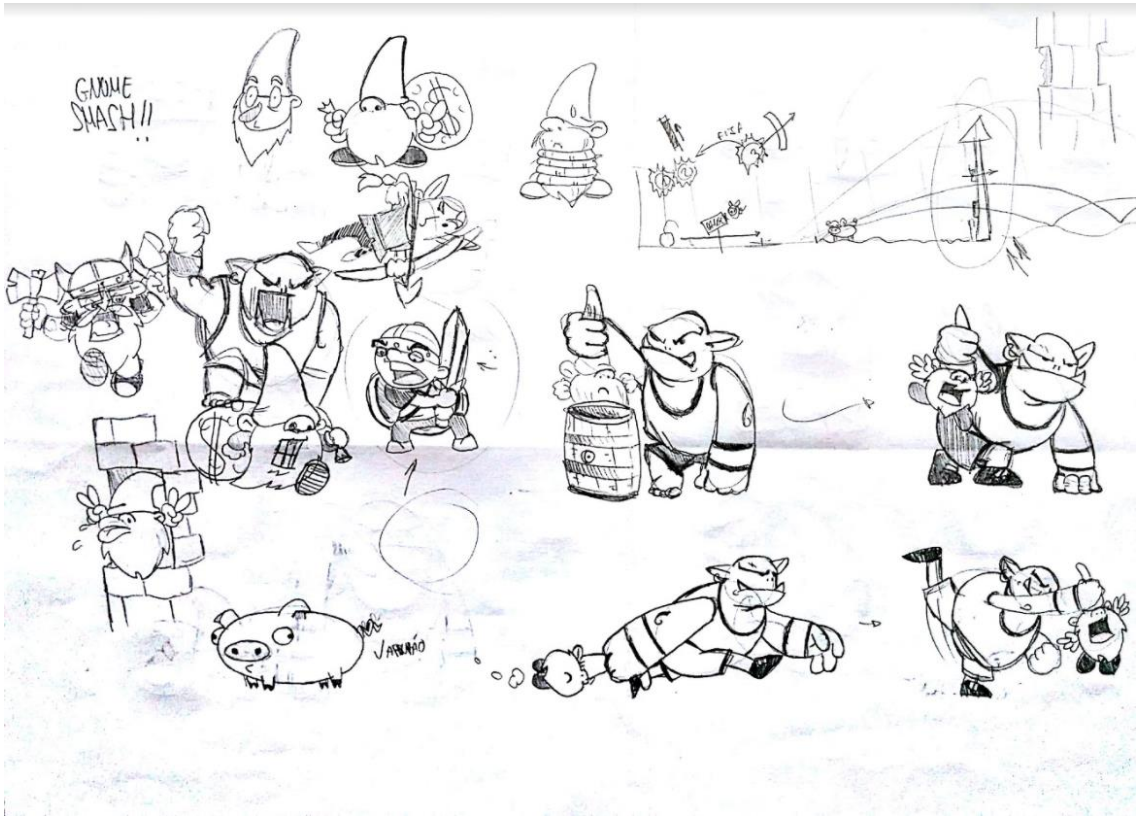

Figura 38: Bocetos iniciales para el videojuego *Gnomore Gnomes*.

Para el recuerdo queda una grabación de un momento del evento que nunca olvidaremos, y que, por ahora, no podemos dejar de ver [12]. Nuestro objetivo estaba cumplido.

A nivel personal, este proyecto me ha permitido enfrentarme cara a cara con el proceso de desarrollo de un proyecto sin guías de cómo proceder o cómo lidiar, evitar y resolver los diversos problemas que iban surgiendo, tanto técnicos como personales. Pero, sobre todo, me ha permitido descubrir que los problemas no son tan problemas cuando las ganas de ver el resultado final son mayores, convirtiendo a los problemas en desafíos.

En cuanto al futuro de *Gnomore Gnomes*, la campaña de *crowdfunding* en cuya preparación nos hallamos inmersos mientras escribimos estas líneas, determinará si podremos seguir trabajando para mejorar y agrandar nuestro proyecto, y llegar así a cualquier jugador en el mundo que simplemente quiera divertirse con nuestra obra. Una vez logrado ese objetivo, el futuro del juego pasa por ampliarlo, lanzarlo en más plataformas, llegar a más público, e incluso plantearse secuelas o diverso contenido relacionado con el universo del juego (como cómics, cortos de animacoión, etc.)

Nos gustaría concluir esta memoria con una cita del ya fallecido Satoru Iwata, antiguo director de Nintendo, un hombre que dedicó su vida a llevar la magia de los videojuegos a todos los jugadores del mundo, y que, probablemente, sea nuestro mayor referente:

> *"Una cosa que no ha cambiado -y no que cambiará- es nuestra propia naturaleza como una forma de entretenimiento. Como cualquier otro medio de entretenimiento, debemos crear una respuesta emocional para poder triunfar. Risas... miedo... gozo... rabia... afección... sorpresa... y sobre todo, orgullo por los logros conseguidos. Al final, generar esos sentimientos en nuestro jugadores es el verdadero juicio de nuestro trabajo. Esta es la forma definitiva de medir el éxito".*
>
> - *Satoru Iwata*

Referencias y Bibliografía

[1] Información sobre la recaudación de ventas de videojuegos
http://roibertjohan.blogspot.com.es/.....8

[2] El videojuego Bioshock 2 incorpora modo multijugador
https://www.destructoid.com/2k-tells-us-why-bioshock-2-s-multiplayer-exists-162744.phtml.....17

[3] El videojuego Mass Effect 3 incorpora modo multijugador
http://www.ign.com/articles/2011/10/10/mass-effect-3-multiplayer-confirmed.....17

[4] El videojuego Tomb Raider incorpora modo multijugador
http://www.ubergizmo.com/2013/01/tomb-raider-to-come-with-multiplayer-after-all/.....17

[5] Las videoconsolas más vendidas
https://en.wikipedia.org/wiki/List_of_best-selling_game_consoles.....18

[6] Información sobre las ventas del videojuego Minecraft
http://www.hobbyconsolas.com/noticias/minecraft-mas-100-millones-copias-vendidas-145454.....19

[7] Información y datos sobre la comunidad de jugadores activa en Steam
https://www.vidaextra.com/industria/steam-consigue-superar-los-125-millones-de-cuentas-activadas.....20

[8] Información sobre la distribucion de videojuegos en Steam en 2014
http://kotaku.com/nearly-40-of-all-steam-games-were-released-in-2016-1789535450.....20

[9] Vídeo “Top 10 videojuegos de 2014 - Post Script”
https://www.youtube.com/watch?v=p31um8D-pAg.....22

[10] Top 50 juegos durante el año 2014
http://www.eurogamer.net/articles/2015-01-02-readers-top-50-games-of-2014 23

[11] Nintendo y el contenido extra en sus videojuegos
http://es.ign.com/mario-kart-8-wii-u/80498/feature/nintendo-y-los-dlc.....24

[12]Vídeo de los usuarios probando Gnomore Gnomes durante el evento B&T

https://www.youtube.com/watch?v=0sxpvJcD4e0.....68

- Barrera, R.; Peters, P.; Sithu, A. y Naing, T. (2015). *Unity AI Game Programming.* Packt Publishing

- Archer, T. (2001). *A fondo C#.* McGRAW-HILL

- Smith, M.y Queiroz, C. (2015). *Unity 5.x Cookbook.* Packt Publishing

Material y transparencias proporcionadas por los cursos realizados en el centro de formación permanente de la Universidad Politécnica de Valencia:

-Curso 3D Studio Max (Autodesk).

-Introducción al desarrollo de videojuegos en Unity 3D.

Índice de Ilustraciones

Anexo: Flujo de una partida en *Gnomore Gnomes*

A continuación, vamos a describir el flujo de una partida del videojuego *Gnomore Gnomes* en su última versión mínima jugable hasta la fecha. Mostramos esquema y una breve explcación del contenido de cada una de las distintas vistas del videojuego y de las transiciones que existen entre ellas:

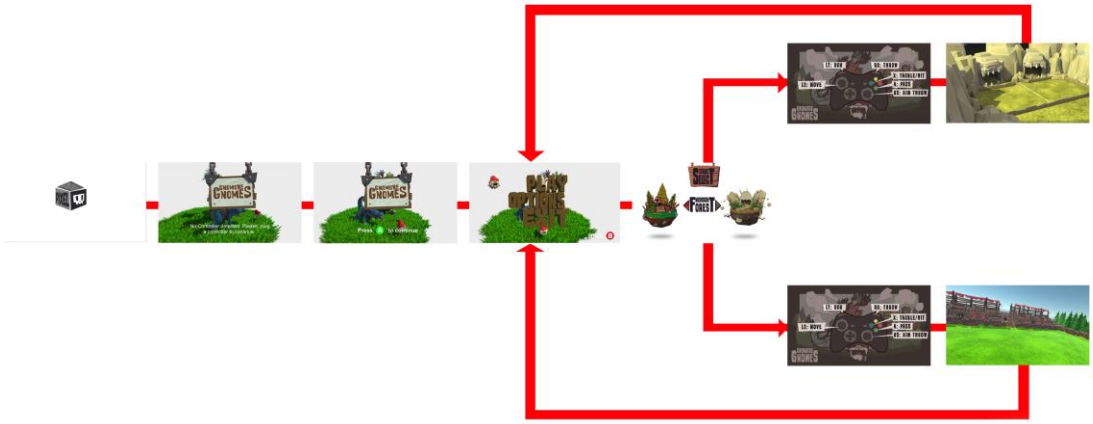

Figura Anexo 1: Vista esquemática del flujo de ejecución de *Gnomore Gnomes*.

Como podemos ver en la imagen anterior, mostramos el flujo de ejecución normal para la versión actual del videojuego *Gnomore Gnomes,* el cual pasamos a detallar a continuación en cada una de las diferentes vistas que ofrece.

Figura Anexo 2: Vista del logo de *Dead Pixel*.

Se muestra durante unos instantes el logo de Dead Pixel. Después de unos segundos el logo desaparece de la pantalla dando paso al inicio del juego.

Figura Anexo 3: Pantalla de detección de controlador de juego.

Se muestra por pantalla la vista de la imagen anterior. Esta vista se mantiene a la espera hasta que un usuario conecta un controlador USB para juegos en uno de los puertos del ordenador.

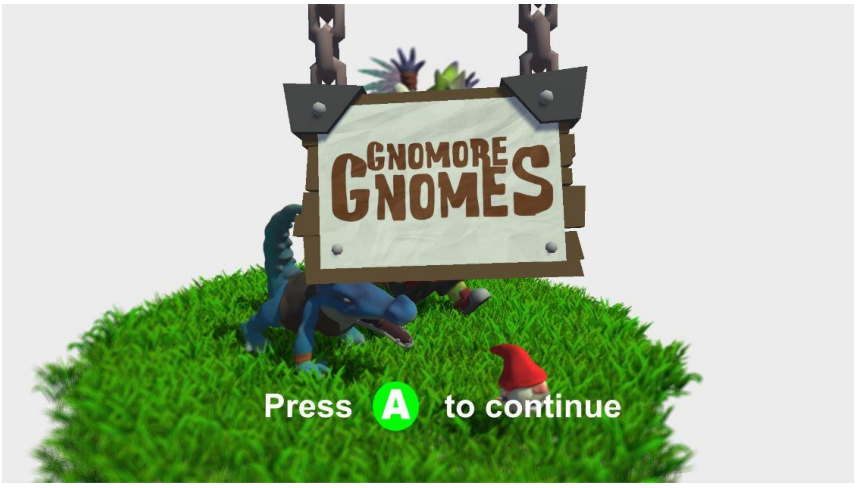

Figura Anexo 3: Pantalla de confirmación de inicio del videojuego.

Una vez conectado un controlador, se pide confirmación al usuario para iniciar el videojuego pidiéndole que pulse el botón “A” o *botón 1* del mando.

Figura Anexo 4: Vista del menu de inicio.

La imagen anterior pertenece a la vista del menú de inicio. Aquí se muestran las diferentes opciones que se le ofrecen al usuario.

- Play: Al pulsar esta opción el usuario podrá elegir el escenario donde desea iniciar la partida para empezar la partida.
- Options: Aparecen las diferentes opciones que permiten al usuario configurar el videojuego asu gusto. El menú de opciones se encuentra actualmente en desarrollo, por lo que no es posible mostrar una imagen de dicha vista.
- Exit: El juego se cierra completamente, dando paso al escritorio de Windows.

Figura Anexo 5: Pantalla de selección de scenario.

Una vez el usuario ha pulsado la opción *Play* se mostrará una vista donde se le dará a elegir a los jugadores el escenario en el que quieren realizar la partida, podrán elegir entre dos por el momento, que son los que están disponibles en la versión actual del videojuego. Al pulsar el botón “A” comenzará la carga de la partida en el escenario previamente seleccionado.

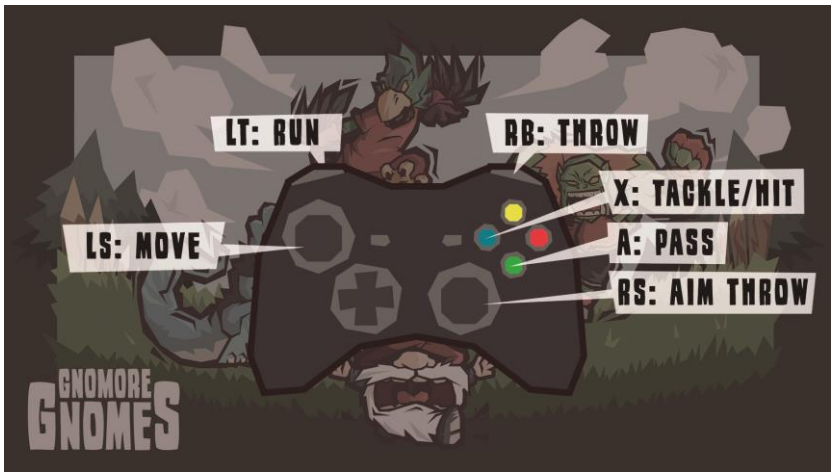

Figura Anexo 6: Pantalla de carga del juego.

Durante el tiempo de carga, se muestra la imagen anterior de modo que el usuario pueda familiarizarse con los controles del videojuego antes de iniciar la partida. Una vez haya cargado esta vista se mantendra durante el tiempo deseado hasta que uno de los usuarios pulse el botón A. A continuación se dará paso al inicio de la partida.

Figura Anexo 7: Vista del scenario *Rock'n Troll Hills*.

En la imagen se muestra *Rock'n Troll Hills,* nombre que recibe el escenario y donde se efectuará la partida si el usuario selecciónó jugar en ese escenario.

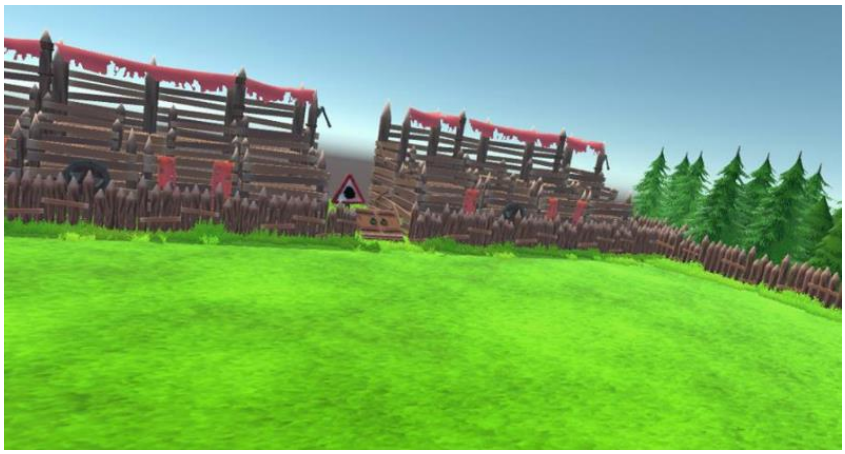

Figura Anexo 7: Vista del scenario *MushBoom Forest*.

En la imagen anterior se muestra *MushBoom Forest,* nombre que recibe este escenario y donde se efectuará la partida si el usuario seleccionó.

Una vez finalizada la partida el videojuego vuelve a la pantalla del menú principal y el flujo de ejecución continúa desde ese punto hasta que el usuario decide dejar de jugar y selecciona la opción *Exit* de dicho menú.