



**UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA**

GRADO EN INGENIERÍA INFORMÁTICA

**TECNOLOGÍA ESPECÍFICA DE
COMPUTACIÓN**

TRABAJO FIN DE GRADO

**Virus Breaker
Desarrollo de un videojuego con Unity3D**

Pedro Romero González

Febrero, 2018

VIRUS BREAKER
DESARROLLO DE UN VIDEOJUEGO CON UNITY3D



**UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA**

Tecnologías y Sistemas de Información

**TECNOLOGÍA ESPECÍFICA DE
COMPUTACIÓN**

TRABAJO FIN DE GRADO

**Virus Breaker
Desarrollo de un videojuego con Unity3D**

Autor: Pedro Romero González

Director: Dr. David Vallejo Fernández

Febrero, 2018

Pedro Romero González

Ciudad Real – España

E-mail: pedro9romero4gonzalez@gmail.com

Teléfono: 689 426 432

Web site: <https://gamejolt.com/@nexus64>

© 2018 Pedro Romero González

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Se permite la copia, distribución y/o modificación de este documento bajo los términos de la Licencia de Documentación Libre GNU, versión 1.3 o cualquier versión posterior publicada por la *Free Software Foundation*; sin secciones invariantes. Una copia de esta licencia esta incluida en el apéndice titulado «GNU Free Documentation License».

Muchos de los nombres usados por las compañías para diferenciar sus productos y servicios son reclamados como marcas registradas. Allí donde estos nombres aparezcan en este documento, y cuando el autor haya sido informado de esas marcas registradas, los nombres estarán escritos en mayúsculas o como nombres propios.

TRIBUNAL:

Presidente:

Vocal:

Secretario:

FECHA DE DEFENSA:

CALIFICACIÓN:

PRESIDENTE

VOCAL

SECRETARIO

Fdo.:

Fdo.:

Fdo.:

Índice general

Índice general	v
Índice de figuras	vii
1. Estado del arte	1
1.1. El mercado de los videojuegos	1
1.1.1. Industria mundial del Videojuego	1
1.1.2. La industria de los videojuegos en España	2
1.1.3. Retos y tendencias actuales	3
1.2. El proceso de desarrollo de videojuegos	11
1.2.1. Introducción	11
1.2.2. Etapas del desarrollo	12
1.2.3. Estructura típica de un equipo de desarrollo	15
1.3. Motores de juegos	22
1.3.1. Descripción	22
1.3.2. Ejemplos de Motores	24
1.4. Inteligencia Artificial en Videojuegos	28
1.4.1. Historia	28
1.4.2. IA Aplicada a Videojuegos: Contexto Actual	29
1.4.3. Métodos de la IA	31
1.4.4. Futuros campos de aplicación	35
Referencias	39

Índice de figuras

1.1. Crecimiento Mundial de la industria del videojuego (por plataformas)	1
1.2. Distribución por regiones del mercado del videojuego.	2
1.3. Diez principales mercados del videojuego.	3
1.4. Mercado del Videojuego en Europa.	4
1.5. Distribución de las empresas en España por número de empleados.	5
1.6. Distribución de las empresas en España por número de empleados.	5
1.7. Fotografía del torneo Intel® Extreme Masters en Katowice, Polonia	6
1.8. Crecimiento del mercado del eSport	6
1.9. League of Legends, uno de los eSports más populares	7
1.10. Previsiones de crecimiento del mercado de Realidad Virtual y Aumentada (miles de millones de dólares))	8
1.11. De izquierda a derecha HTC Vive, Oculus Rift, Samsung VR y PlayStation VR	9
1.12. Áreas de la Industria 4.0	10
1.13. Impresora 3d “replicator” de la compañía Makerbot	11
1.14. Etapas del desarrollo de un videojuego.	12
1.15. Distribución de roles en un equipo de desarrollo.	16
1.16. Shigeru Miyamoto (creador de Super Mario, Zelda y Donkey Kong) es posiblemente el diseñador de videjuegos más famoso	17
1.17. Jonh Romero es el co-fundador de ID Software y programador de juegos como Doom o Quake	18
1.18. Akira Toriyama, el autor de Dragon Ball, ha trabajado como artista en juegos como Dragon Quest o Chrono Trigger	19
1.19. Yoko Shinomura es una compositora conocida por su trabajo en videojuegos como Final Fantasy o Street Fighters	20
1.20. Heretic (Raven Software, 1994), es un juego desarrollado con el motor de Doom.	22
1.21. Captura del entorno de desarrollo de Unity	25
1.22. A Hat in Time (Gears for Breakfast, 2017)	26
1.23. Hollow Knight (Team Cherry, 2017)	26

0. ÍNDICE DE FIGURAS

1.24. Juegos desarrollados con Unity3D	26
1.25. Captura del entorno de Unreal Engine	26
1.26. Unreal (Epic Games, 1998)	27
1.27. Batman: Arkham Knight (Rocksteady Studios, 2015)	27
1.28. Juegos desarrollados con Unreal Engine	27
1.29. Captura del entorno de Game Maker Studio	28
1.30. Undertale (Toby Fox, 2015)	28
1.31. Hyper Light Drifter (Heart Machine, 2016)	28
1.32. Juegos desarrollados con Game Maker	28
1.33. El Gran Maestro de ajedrez Garri Kaspárov (izquierda) enfrentándose al ordenador Deep Blue	30
1.34. En Los Sims (Maxis, 2000-2014), los personajes pueden tomar decisiones basándose en sus gustos y necesidades.	31
1.35. Minecraft (Mojam, 2009) es un ejemplo claro de generación procedimental de terrenos.	32
1.36. Ejemplo de comportamiento tóxico en League of Legends(Riot Games, 2009).	36

Capítulo 1

Estado del arte

1.1 El mercado de los videojuegos

1.1.1 Industria mundial del Videojuego

El mercado del videojuego es actualmente la industria de ocio y entretenimiento más grande, superando en tamaño tanto a la industria cinematográfica como a la discográfica. Se trata de una industria creciente, con una tasa del crecimiento del 6,6 % y que cuenta ya con más de 2.200 millones de jugadores en todo el mundo. Se espera que, en el año 2020, la cotización anual alcance los 143.500 millones de dólares [DEV17].

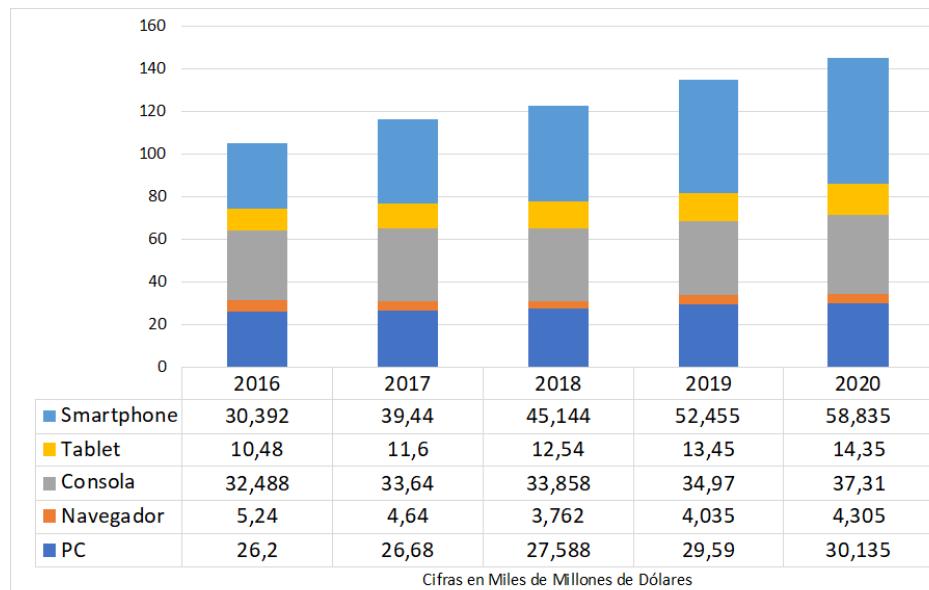


Figura 1.1: Crecimiento Mundial de la industria del videojuego (por plataformas).

Actualmente, la plataforma de distribución que ocupa un segmento mayor del mercado son los dispositivos móviles. Debido al constante incremento de potencia de los Smartphones, así como a su ubicuidad en la sociedad actual, el mercado de videojuegos para estas plataformas ha experimentado un incremento constante en los últimos años, superando al mercado para PC y al mercado para Videoconsolas de sobremesa. A fecha de 2017, el mercado Smartphones ha alcanzado los 39.440 millones de dólares, lo que representa el 34 % del mercado. Por otro lado, el mercado de las consolas portátiles y el de los juegos Web casuales son los que presentan un declive más pronun-

1. ESTADO DEL ARTE

ciado, debido seguramente a que ocupan un nicho de mercado similar al de los juegos móviles[DEV17].

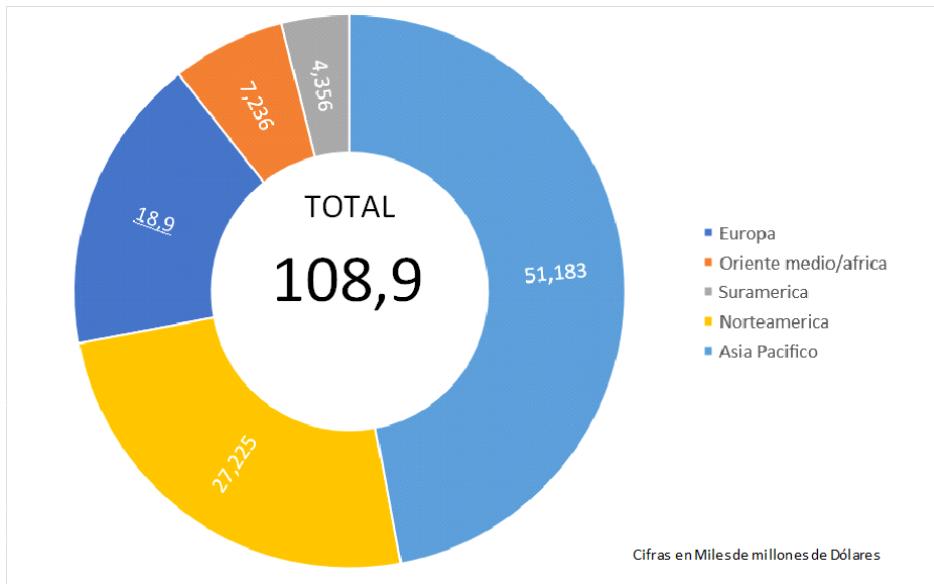


Figura 1.2: Distribución por regiones del mercado del videojuego.

Si dividimos el mercado en las distintas regiones geográficas, podemos observar que Asia Pacifico lidera el mercado con un beneficio de 51.200 millones de dólares en el año 2017, un 47 % del total de los ingresos globales, de los cuales más de la mitad son generados por China, con 227.500 millones de dólares. En segunda posición se encuentra el mercado norteamericano, con 23.500 millones de dólares en 2016, representado prácticamente en su totalidad por Estados Unidos. En el ranking de los 10 mercados con mayores ingresos podemos encontrar cinco países europeos: Alemania, Reino Unido, Francia, España e Italia. Sin embargo, la diferencia en tamaño con respecto a los tres primeros mercados de la lista (China, Estados Unidos y Japón) es abismal[DEV17].

1.1.2 La industria de los videojuegos en España

La industria del videojuego española es la cuarta mayor de Europa (por detrás de Alemania, Reino Unido y Francia) y la novena mayor a nivel mundial. A fecha de 2017, el mercado español del videojuego facturó un total de 1.900 millones de dolares, con un crecimiento del 20 % con respecto al año anterior. Más de la mitad de estos ingresos provienen de la venta de videojuegos españoles al extranjero, gracias a la casi total falta de fronteras para la distribución internacional de productos. Este enfoque en el mercado internacional se ve reflejado en factores como la mayor frecuencia del inglés en las producciones españolas que el propio español (99 % contra 95 %)[DEV17].

En total, el sector cuenta con 480 empresas en activo, a las que debemos añadir las 130 iniciativas y proyectos empresariales, que se encuentran a la espera de consolidarse como empresas en el corto o medio plazo. La mayor parte de estas empresas tienen

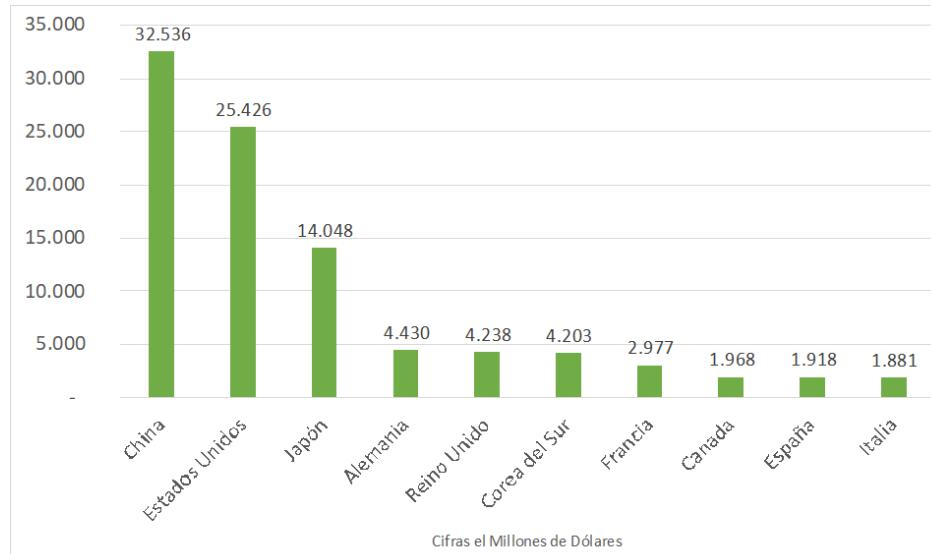


Figura 1.3: Diez principales mercados del videojuego.

una plantilla de menos de 5 empleados, formando el 47% de la industria. Esto se debe en parte a la adecuación de las pequeñas empresas a la creación de juegos de pequeña escala para dispositivos móviles (el principal mercado), pero también se debe a una escasez de puestos de trabajo en las empresas de tamaño mediano y grande y a la saturación del mercado que dificulta el crecimiento de las empresas[DEV17].

La actividad empresarial del país se encuentra centrada en dos comunidades autónomas: Cataluña y la comunidad de Madrid. De estos dos centros principales destaca Cataluña, donde se concentra el 52% de la facturación del país. Detrás de las dos comunidades principales se encuentran la Comunidad Valenciana, el País Vasco y Andalucía, las cuales suman entre las tres un 28% de las empresas. El resto de comunidades se quedan muy por detrás de estas cinco primeras[DEV17].

1.1.3 Retos y tendencias actuales

El videojuego ha sido y es una industria muy cambiante, que siempre ha intentado integrar las tecnologías más punteras, desde innovadores algoritmos de renderizado gráfico hasta exóticos dispositivos de interacción persona-ordenador.

A continuación, listaremos algunas de las tendencias que van a influir fuertemente en el mercado en los años venideros:

eSports

Los eSports, también llamados “deportes electrónicos”, es el nombre por el cual se conocen las competiciones de videojuegos multijugador. En los eSports, los jugadores profesionales compiten entre ellos en diferentes categorías: disparos en primera persona, lucha, estrategia en tiempo real, MOBAs (Multiplayer Online Battle Arena)... La

1. ESTADO DEL ARTE

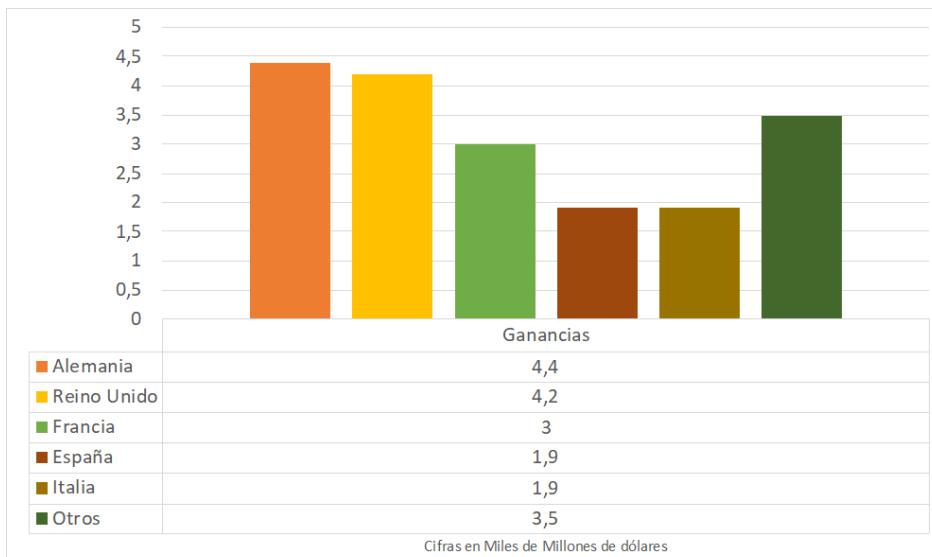


Figura 1.4: Mercado del Videojuego en Europa.

popularidad de este fenómeno ha llegado al punto en el que los principales torneos se celebran en grandes estadios, están retransmitidos en streaming por Internet e incluso están dotados con premios de grandes cantidades de dinero y que en ocasiones superan el millón de euros. Se trata de unos eventos de gran popularidad que cuentan enormes con perspectivas de crecimiento (a una tasa anual del 34 %). Esto ha provocado que se haya posicionado como un fenómeno de ocio estratégico[DEV17].

El mercado de los eSports, generó en 2017 600 millones de dólares de ingresos, creciendo del 67 % con respecto al año anterior. Si la situación se mantiene, se espera que la cifra supere los mil millones de dólares en el año 2019, manteniendo un crecimiento anual superior al 34 %. En 2016, hubo 385,5 millones de personas que vieron partidos de competiciones de diversos deportes electrónicos, de los cuales 115 millones pueden ser clasificados como “entusiastas”, es decir, espectadores regulares y participantes no muy distintos de los que encontraríamos en los deportes convencionales[DEV17].

Existen en España diferentes empresas que en la actualidad apuestan por el desarrollo de productos que aspiran a convertirse en eSports, es el caso de Digital Legends¹, Mercury Steam², PixelCream Studio³, Mechanical Boss⁴, entre otros. Sin embargo, en la mayoría de las ocasiones un videojuego online se convierte en eSports de manera orgánica, basada en el reconocimiento que la comunidad online de jugadores activos de ese videojuego le otorga. Existen eso si casos en las que una gran marca apuesta porque su producto se convierta en un eSports, realizando grandes inversiones en infraestructura, personal dedicado a la comunidad, servidores escalables para una gran

¹<http://www.digital-legends.com/>

²<https://www.mercurysteam.com/>

³<http://pixelcreamstudio.com/>

⁴<http://www.mechanicalboss.com/>

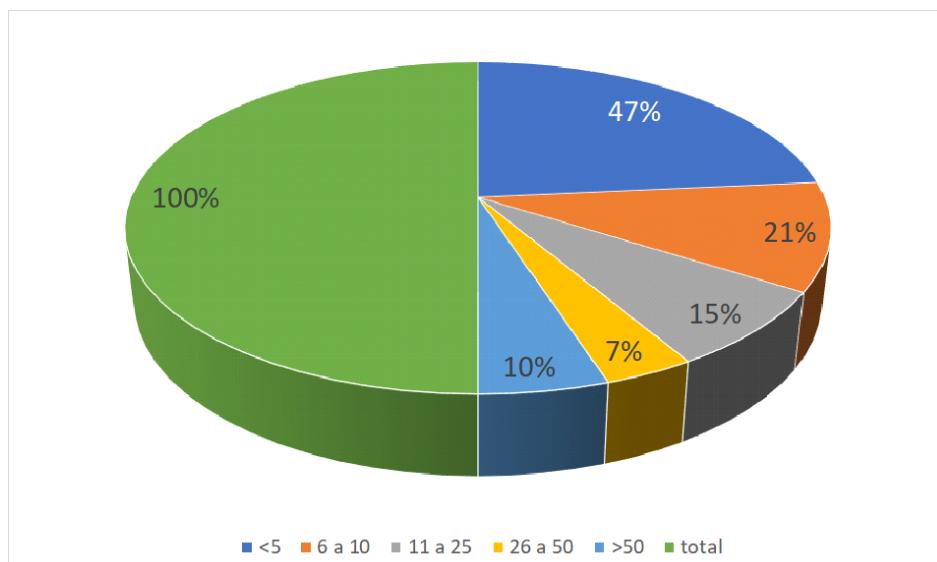


Figura 1.5: Distribución de las empresas en España por número de empleados.

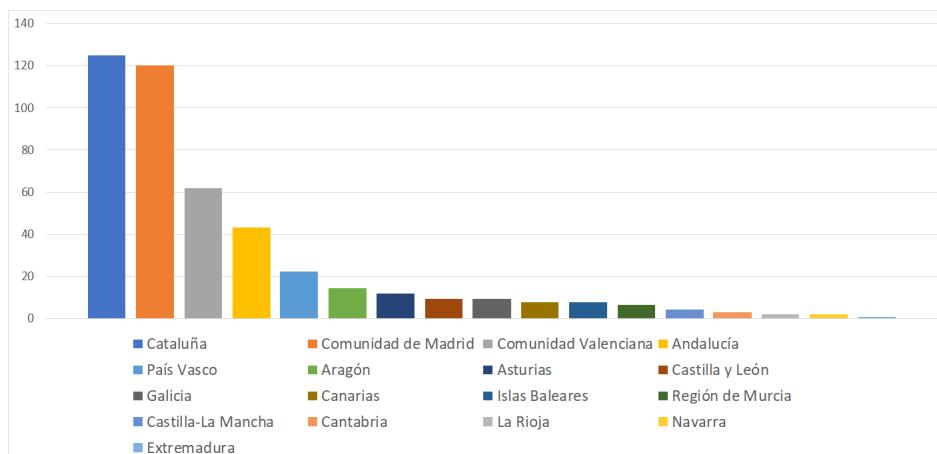


Figura 1.6: Distribución de las empresas en España por número de empleados.

masa de jugadores, premios para los torneos, entre otras muchas.

Sin embargo, esta inversión no siempre asegura que su producto se convierta en un éxito. Para que un videojuego pueda convertirse en un eSport necesita contar con características básicas: tener un fuerte factor de competición, partidas cortas de no más de 1 hora, sin progresión in-game (la progresión debe basarse en las habilidades del jugador) atractivo sistema de espectador y tener un enfoque al 100 % internacional.

Pese a su gran dificultad, conseguir posicionar un producto como eSports, aporta una serie de beneficios y posibilidades:

- Crear una base de fans, una comunidad, algo que aporta un núcleo de consumidores fieles al producto y que le da una nueva dimensión social, muy atractiva para muchos de los consumidores de videojuegos.
- Prolongar la vida del producto; al ser competitivo, el jugador fija sus metas ante

1. ESTADO DEL ARTE



Figura 1.7: Fotografía del torneo Intel® Extreme Masters en Katowice, Polonia

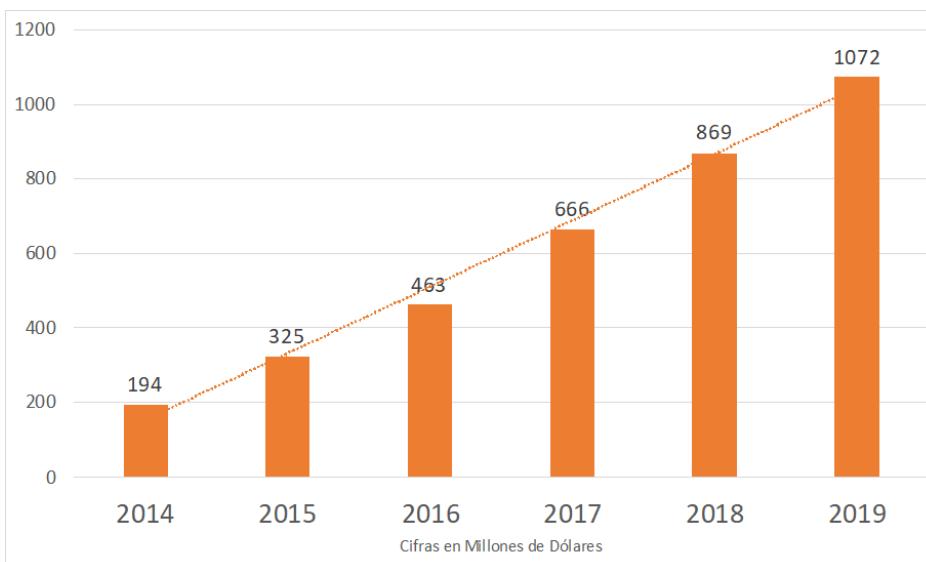


Figura 1.8: Crecimiento del mercado del eSport

los otros jugadores, esto incentiva al usuario y le proporciona una motivación para seguir consumiendo.

- Proporcionar mayor visibilidad, ya que a pesar de que los productos asentados son extremadamente sólidos, su numero es muy reducido, por lo cual hay una demanda latente de usuarios que buscan nuevos eSports.
- Aumentar la fidelidad de los usuarios al tratarse de un mercado donde los usuarios tienen un índice de fidelidad mucho más alto que en otros.
- Los jugadores, al estar involucrado con un producto competitivo, ven streaming, leen noticias, siguen torneos, participan en foros, lo que disminuye el riesgo de abandono del producto.



Figura 1.9: League of Legends, uno de los eSports más populares

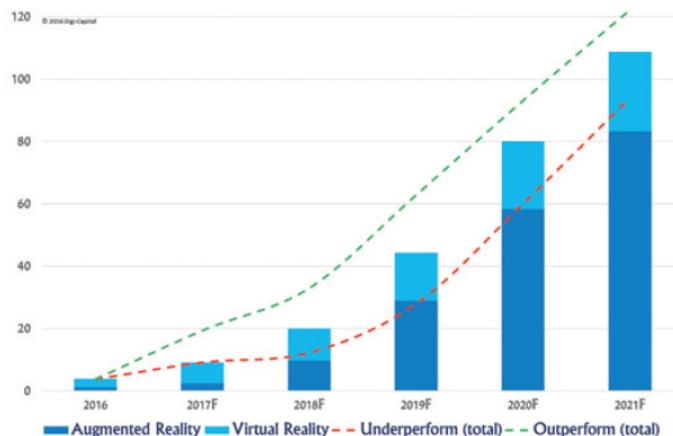
Para una empresa pequeña, la producción de un eSports es, en principio, inabarcable. Esto se debe principalmente la elevada inversión mencionada anteriormente. Sin embargo, la asociación con unos los partners correctos, ya establecidos en el sector, que pudieran invertir en el producto y le den visibilidad, puede dar una gran oportunidad a los pequeños desarrolladores que cuentan con una mayor flexibilidad con respecto a las grandes compañías, lo que les permite adaptarse más rápidamente a un mercado tan cambiante y mantener una relación directa con el feedback del usuario.

Realidad Virtual y Realidad Aumentada

La realidad virtual (normalmente abreviada como VR por las siglas inglesas de Virtual Reality) es la tecnología generada por sistemas informáticos que proporcionan un entorno audiovisual en 3D el que el usuario puede experimentar una inmersión total. Para ello, se hacen usos de cascos especiales equipados con pantallas y sensores de movimiento, los cuales normalmente se complementan con mandos equipados también con sensores para permitir una interacción más natural. Por otro lado, la Realidad Aumentada o AR es una tecnología que superpone una capa de gráficos generados por ordenador sobre el entorno que rodea al jugador, con la que este puede interaccionar en tiempo real. A diferencia de la VR, no se requiere obligatoriamente de un hardware especial para poder implementar AR; basta únicamente de un dispositivo equipado con una pantalla y una cámara de vídeo, como podría ser un Smartphone.

Actualmente, existen varias propuestas de diversas compañías en lo que a equipo de VR se refiere. Vamos a mencionar algunas de las más importantes:

1. ESTADO DEL ARTE



Fuente y gráfico original: Digi-Capital

Figura 1.10: Previsiones de crecimiento del mercado de Realidad Virtual y Aumentada (miles de millones de dólares))

- **HTC Vive⁵:** es la propuesta de HTC y Valve, orientada a jugadores “hardcore” de PC. Disponible desde abril de 2016, el dispositivo requiere de un PC de gama alta (Valve recomienda un PC con una gráfica GeForce GTX 970). El kit de hardware incluye el casco equipado con dos pantallas de 1080x1200 puntos y 90Hz de frecuencia de actualización, dos sensores espaciales y dos mandos para registrar los movimientos de ambas manos, lo que crea le permite crear un entorno 100 % virtual en el que sumergir al jugador.
- **OCULUS Rift⁶:** Es la propuesta más veterana de la lista. Empezó como un exitoso proyecto de Kickstarter en 2012 que más tarde fue adquirida por la empresa Facebook dos años más tarde. Al igual que HTC Vive, Oculus está formado por un casco equipado con pantallas de alta resolución, mandos con sensores de movimiento y dos sensores de posición. El equipo necesita estar conectado a un PC de alta gama para poder funcionar correctamente.
- **Samsung Gear VR⁷:** La propuesta de Samsung es mucho más sencilla y económica, orientado más a la reproducción de vídeo en 360º (concepto similar a la realidad virtual pero con interactividad limitada). El casco incluye una única pantalla y sus mandos carece de detección de movimiento. Estas limitaciones conllevan, por otro lado, un precio mucho más accesible que el de las otras alternativas (99€ contra los más de 500€ de las propuestas más completas)
- **SONY PlayStation VR⁸:** la propuesta de Sony fue lanzada en el año 2016. Al igual que otras alternativas, el sistema se basa en un casco equipado con

⁵<https://www.vive.com/>

⁶<https://www.oculus.com/rift/>

⁷<http://www.samsung.com/es/wearables/gear-vr-sm-r325nzbaphe/>

⁸<https://www.playstation.com/explore/playstation-vr/>

dos pantallas y sensores de movimiento, pero su principal punto de venta es su compatibilidad con la consola PlayStation 4 de la misma marca. Esto permite aprovechar la potencia y los mandos de control de ésta de la consola.



Figura 1.11: De izquierda a derecha HTC Vive, Oculus Rift, Samsung VR y PlayStation VR

Web 4.0

El término Industria 4.0 fue acuñado por el Ministerio de Educación y Desarrollo alemán en su plan estratégico de 10 puntos del año 2016 para mejorar la educación, investigación e industria del país para adaptarlas a las tecnologías de Internet [Lyd]. La estrategia trata cinco áreas principales:

- Fuerte cooperación entre la investigación científica y las empresas.
- Aumentar la innovación en el sector privado.
- Diseminar las tecnologías punteras.
- Internacionalizar la investigación y desarrollo.
- Fondos para individuos con talento.

De entre las distintas tecnologías que podrían categorizarse como parte de la industria 4.0, vamos a describir aquellas que tienen mayores aplicaciones en el desarrollo de videojuegos:

La computación en la nube, o Cloud Computing, es la tecnología que permite el acceso a servicios informáticos de forma rápida y sencilla a través de Internet. Aunque aún no se ha podido implementar correctamente el Cloud Gaming (donde el juego es íntegramente ejecutado en la nube, reduciendo la exigencia de potencia del sistema del jugador), si se utilizan sistemas en la nube en distintas áreas de los videojuegos. Especialmente notable es su uso para el control y almacenamiento de información en juegos multijugador en línea.

El Internet de las cosas es como se conoce a la tecnología que permite dotar de conexión a Internet a todo tipo de pequeños dispositivos como relojes, sistemas de domótica, drones, sensores de todo tipo, robots, etc. Esto permite implementar videojuegos en todo tipo de sistemas, desde consolas portátiles cada vez más pequeñas y económicas, pasando por juguetes interactivos y llegando a la posibilidad de gamificar con facilidad procesos industriales.

PRINCIPALES ÁREAS DE LA INDUSTRIA 4.0



Figura 1.12: Áreas de la Industria 4.0

Big Data es el proceso de clasificar grandes volúmenes de datos para poder obtener relaciones interesantes y no evidentes entre ellos. El principal uso de las técnicas de BigData en la industria del Videojuego es el análisis de la información de los jugadores. Analizando datos de los jugadores tales como el género, la edad, la localización geográfica, los intereses, los gastos realizados, etc. es posible obtener estrategias de negocios eficientes.

Los sistemas Ciberfísicos son un nuevo tipo de sistemas con unos componentes hardware y software estrechamente interconectados, cada uno operando en su propio ámbito, operando e interaccionando de forma distinta dependiendo del contexto[SKK14]. Entre sus aplicaciones se encuentran las redes eléctricas inteligentes, los sistemas de conducción automática de aviones y automóviles o la monitorización médica. Las interfaces de estos sistemas requieren de unas interfaces con un fuerte “lado humano” que permita un uso sencillo e intuitivo. Aquí se podrían utilizar los principios de diseño de juego que

permitirían desarrollar un mejor puente entre el lado máquina y la parte de usuario.

La Impresión 3D, también conocida como la producción aditiva es una tecnología que permite producir objetos de forma más sencilla que con las técnicas anteriores. En combinación con las técnicas de escaneado 3D, los estudios de videojuego pueden generar de forma rápida y eficiente modelos 3D de todo tipo (personajes, mapas, objetos...).



Figura 1.13: Impresora 3d “replicator” de la compañía Makerbot

Las nuevas tecnologías de la industria 4.0 serán de gran ayuda para el desarrollo de videojuego. Pero es posible que la industria 4.0 también ofrezca valor a la industria 4.0 en su conjunto. Dado que la creación de videojuegos es una actividad industrial que está vinculada a diferentes áreas de conocimiento que trabajan juntas para conseguir ofrecer un producto, las técnicas y paradigmas utilizados tienen mucho en común con la nueva forma de trabajar de la industria 4.0, por lo que es posible que puedan extrapolarse a otras industrias, permitiendo una mejor adaptación a los cambios.

1.2 El proceso de desarrollo de videojuegos

1.2.1 Introducción

El desarrollo de un videojuego, como el de cualquier otro producto software, debe de ser planificado correctamente y ejecutado siguiendo una metodología adecuada. Sin embargo, El diseño y desarrollo de un videojuego requiere de la participación de campos ajenos a la informática como el diseño de juegos, el diseño gráfico o la composición musical. Una parte importante de la producción consistirá en organizar a un equipo multidisciplinario para poder terminar el proyecto dentro del tiempo y presupuesto acordados [DV15].

1. ESTADO DEL ARTE

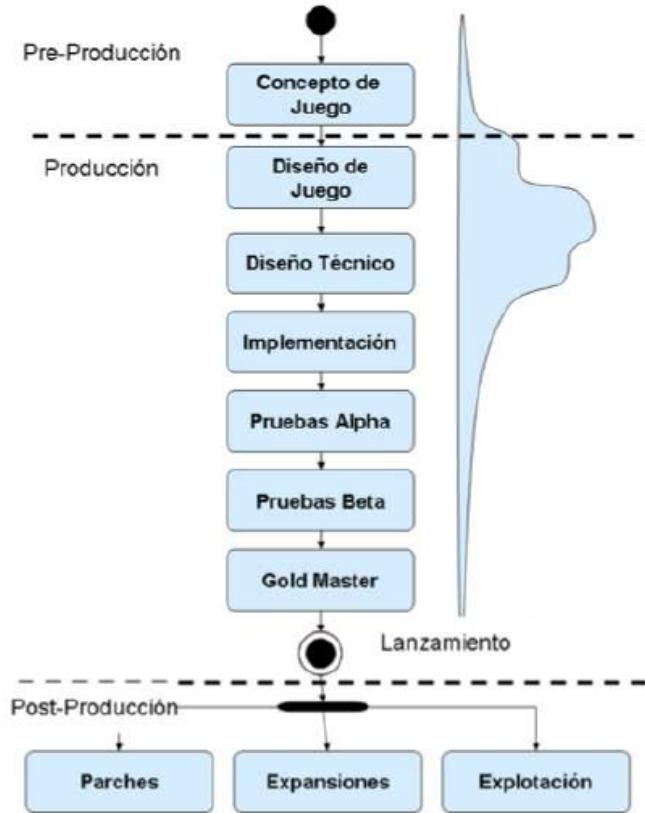


Figura 1.14: Etapas del desarrollo de un videojuego.

1.2.2 Etapas del desarrollo

Desarrollo del Concepto

El desarrollo de todo videojuego comienza con una idea. Durante la fase de desarrollo del concepto se tomará dicha idea para obtener un diseño preliminar listo para preproducción. El objetivo principal de esta etapa es decidir sobre qué tratará el juego y ponerlo por escrito para que cualquier miembro del equipo lo pueda entender con claridad, decidiendo las principales mecánicas, creando el arte conceptual y escribiendo el argumento [Bat04].

Al final de la etapa se habrán elaborado tres documentos: El *High Concept*, el *Pitch Document* y el *Concept Document*. El *High Concept* consiste en una o dos frases que describen a grandes rasgos cómo será el juego, en especial que lo hace distinto de la competencia. El *Pitch Document*, o Propuesta de Juego, es un pequeño documento de en torno a dos páginas, orientado a ser leído en las reuniones donde el juego sea propuesto a inversores. El documento resume las características del juego y explica por qué será exitoso y rentable.

Finalmente, el *Concept Document* es un extenso documento que explica en detalle las características del juego. Se trata de una versión extendida de *Pitch Document* que

trata temas como:

- El High Concept
- El género del juego
- Características principales y jugabilidad.
- Ambientación e historia.
- Estimación del presupuesto y de la planificación.
- Equipo de desarrollo.
- Análisis de Riesgos.

Pre-Producción

La pre-producción es la fase de preparación: en la que el equipo diseña y planifica los elementos que serán desarrollados en la etapa de producción. Al final de esta etapa, se debe haber completado el diseño de juego, creado la biblia de arte, elaborado el documento de diseño técnico, establecido el plan de producción, y creado un prototipo del juego final[Bat04].

El diseño del juego quedará establecido en un Documento de Diseño de Juego (o GDD, por sus siglas en inglés). Se trata de un documento “vivo”, que está en constante modificación para adaptarse a los ajustes concretos de diseño que se realizarán durante el desarrollo.

La biblia de arte es una colección de arte conceptual que servirá para definir el estilo artístico del juego desde un primer momento. La biblia incluirá también una librería de imágenes de referencia que puedan ser de ayuda a los artistas que desarrollen los elementos gráficos finales.

El Documento de Diseño Técnico contiene una descripción en detalle la parte técnica del proyecto definiendo las tareas que los desarrolladores deberán afrontar, estimando el coste que dichas tareas tendrán tanto en tiempo como en número de personas y especificando las herramientas y técnicas que utilizará el equipo.

El Plan de producción recopila la información acerca de cómo se va a desarrollar el proyecto. Este incluye las tareas a realizar junto a los tiempos, costes y dependencias de estas, divididos en varios documentos menores para poder ser organizado mejor:

- **Plan de mano de obra:** Listado del personal, sus horarios y su salario.
- **Plan de recursos** Estimación del coste los recursos externos al proyecto (música, arte, herramientas...)
- **Documento de seguimiento:** Documento donde se realiza un control de los tiempos y plazos del proyecto.

1. ESTADO DEL ARTE

- **Presupuesto:** Contiene el coste mensual del proyecto y el cálculo del presupuesto general
- **Ganancias y Pérdidas:** Estimación de las ganancias y pérdidas del proyecto. Debe ir actualizándose según se avanza en el desarrollo.
- **Definición de Hitos:** Lista de las distintas “metas” del proyecto, que son puntos del desarrollo donde se habrá terminado una cantidad de trabajo importante.

Una vez diseñado y planificado el proyecto, el equipo empezara a trabajar en la creación de un Prototipo. Un prototipo es una pieza de Software funcional que contiene una pequeña fracción del software final. El desarrollo prototípico servirá para varias funciones: poner a prueba el diseño de juego, concretando de forma más precisa la jugabilidad; realizar un simulacro de desarrollo para determinar las dinámicas del equipo y producir una muestra del juego final a inversores y publicadores.

Producción

La producción es la etapa principal del desarrollo del juego. Durante esta etapa se elaborarán e implementarán los elementos descritos y diseñados durante la pre-producción: los programadores implementarán los sistemas del juego, los artistas elaboraran los gráficos definitivos, los diseñadores crearán misiones y niveles, etcétera. Dependiendo del juego en cuestión, una producción normal suele durar entre seis meses y dos años, aunque el desarrollo de juegos pequeños para, por ejemplo, dispositivos móviles puede realizarse en menos tiempo aún.

La etapa de producción es de naturaleza iterativa: El juego va construyéndose en varias etapas en las que se implementan pequeñas porciones de este. Entre etapa y etapa, el juego pasa por un proceso de pruebas en la que se verifica su usabilidad y robustez frente a fallos. Los resultados de las etapas se utilizan como base para recalcular los tiempos y presupuestos de las etapas posteriores. Dividir el desarrollo de esta forma hace que sea más sencillo afrontar problemas y contratiempos que el equipo podría encontrar en las etapas tardías.

Final de Producción y Lanzamiento

Durante las últimas etapas de la producción, el paradigma de desarrollo cambia, pasando el objetivo de implementar contenido nuevo a pulir y ajustar el contenido preexistente. Esta parte de la producción puede dividirse en dos etapas: Alpha y Beta.

La fase **Alpha** o Code-Complete es el punto del desarrollo donde el juego se encuentra en un estado jugable, a falta solo de ciertos vacíos como gráficos provisionales o mini juegos o sub-sistemas incompletos. El objetivo de esta etapa es el de encontrar y corregir todos los fallos posibles y también probar y ajustar la jugabilidad[Bet03].

En la fase **Beta** o Content-Complete la mayor parte del contenido del juego deberá estar terminado y debe haber pocos o ningún fallo importante en el juego. En esta etapa el juego es puesto en manos de equipos de testing externos a la empresa para realizar análisis exhaustivos en busca de fallos que se le pueden haber escapado al equipo de testing interno. Es también en esta etapa donde la campaña de publicidad del juego deberá ser más fuerte[Bet03].

Una vez superadas las dos etapas de pruebas, la Versión Final del juego es enviada a la distribuidora para que comience la producción de las copias físicas, o para que el juego aparezca en las plataformas de distribución.

Post-Producción

Una vez lanzado el juego, el equipo entra en la etapa de Post-Producción. En esta etapa el equipo trabajará en corregir fallos y problemas que los jugadores encontraron tras el lanzamiento y en la elaboración de contenido adicional descargable.

La duración y trabajo de la post-producción depende mucho del juego en cuestión. Hasta hace relativamente poco, los juegos lanzados en videoconsolas carecían completamente de esta etapa debido a la dificultad para modificar los juegos que ya se encontraran en el mercado. por otro lado, hoy en día la mayor parte de los videojuegos reciben parches y actualizaciones sin importar su plataforma de distribución[Bet03].

Un caso especial sería el de los juegos con un fuerte componente Online, como los MMORPG, los MOBA o los shooter en línea. Los desarrolladores de este tipo de juegos, para mantener contenta a su base de jugadores y evitar el estancamiento, lanzan de forma periódica actualizaciones que ofrecen nuevo contenido, mejoras y ajustes. Algunos de estos juegos pueden recibir este tipo de actualizaciones durante años, como World of Warcraft⁹ o Team Fortress 2¹⁰ (FPS de Valve, en activo desde 1999)

1.2.3 Estructura típica de un equipo de desarrollo

El desarrollo de un videojuego puede llevarse a cabo por equipos de desarrollo muy distinto dependiendo de la extensión del proyecto, desde una sola persona creando un pequeño juego independiente, el cual realiza todo el trabajo por sí mismo; hasta los equipos de cientos de personas que desarrollan los juegos “triple A”, organizados en múltiples departamentos, cada uno con su estructura jerárquica.

A pesar de esto, existen una serie de roles que están presentes en todos los desarrollos. En los proyectos pequeños será normal encontrar que una misma persona ejerce varios roles, mientras que las grandes compañías pueden tener departamentos enteros encargándose de un único rol.

⁹<https://worldofwarcraft.com> (MMORPG de Blizzard, en activo desde 2004)

¹⁰teamfortress.com/

1. ESTADO DEL ARTE

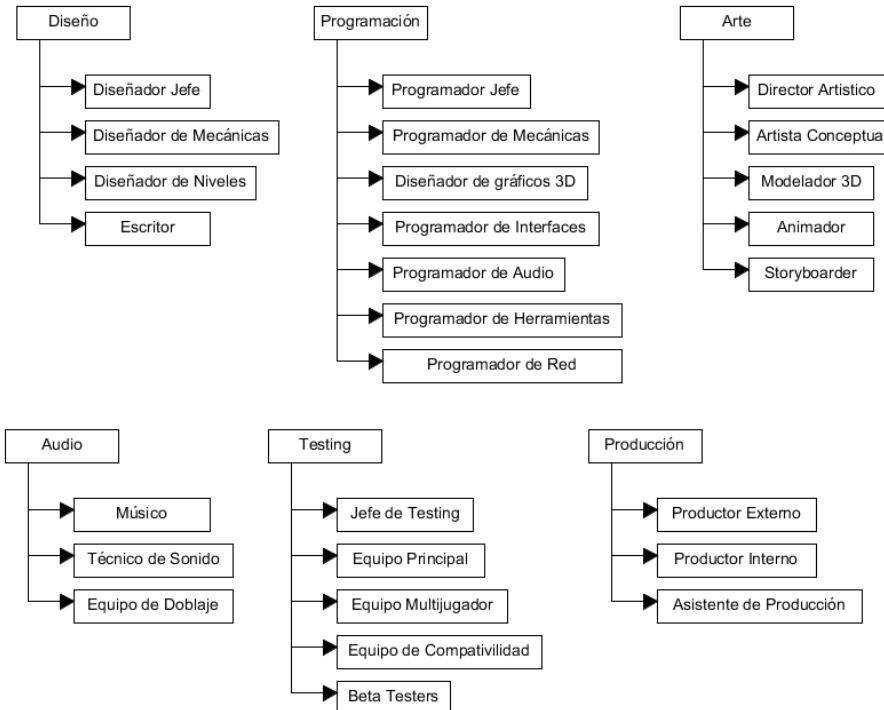


Figura 1.15: Distribución de roles en un equipo de desarrollo.

Diseño

La primera y más importante parte del desarrollo de un juego es el Diseño de este. El trabajo del diseñador es de describir el juego con un alto nivel de detalle, definiendo de con precisión todas las mecánicas, personajes, mapas, misiones del juego. Deberá también, hasta cierto punto, coordinar y dirigir el trabajo del resto de miembros del equipo para que puedan implementar correctamente el juego.

En equipos grandes, el rol de diseñador puede dividirse en las siguientes categorías[Bet03]:

1. **Diseñador Jefe:** Es el encargado de dirigir al resto de diseñadores, decidiendo que contenido entra o no en el juego. Suele ser la persona que tuvo la “idea” original del juego.
2. **Diseñador de mecánicas:** El diseñador de mecánicas es el encargado de diseñar los distintos sistemas de juego, sirviendo como puente entre el diseñador jefe y los programadores. Debido a esto, el diseñador de mecánicas suele tener un trasfondo de programador.
3. **Diseñador de niveles:** También llamados diseñadores de misiones, son los encargados de crear las distintas etapas que componen el juego, ya sean niveles, misiones, desafíos o puzzles.
4. **Escritor:** La tarea del escritor es la de crear la historia del juego, así como la de escribir los distintos textos de este, como diálogos o descripciones. Se trata de

una tarea muy distinta de la de un escritor de novelas o de guiones de película, ya que debe conciliar la narrativa con las exigencias de otros componentes del juego como el diseño, el arte o las limitaciones técnicas.

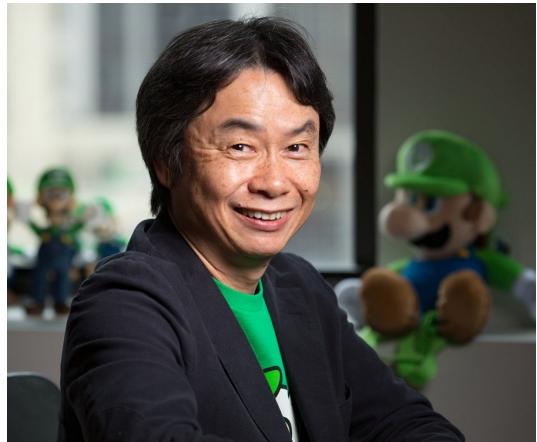


Figura 1.16: Shigeru Miyamoto (creador de Super Mario, Zelda y Donkey Kong) es posiblemente el diseñador de videjuegos más famoso

Programación

El rol del programador es el de implementar el juego en forma de código ejecutable. Esto supone el diseño e implementación de todo tipo de componentes imprescindibles: motor de renderizado, librerías para trabajar con sistemas de audio o conectarse por Internet, herramientas para integrar fácilmente el contenido artístico entre otras.

Cuando el equipo de desarrollo es grande, el rol de programador tiende a dividirse para cubrir tareas más específicas[Bet03]:

1. **Programador Jefe:** El programador jefe es frecuentemente el programador con más experiencia del equipo y él es el encargado de resolver las tareas más complicadas e importantes del proyecto. Cuando el equipo es muy grande, suelen realizar también tareas de coordinación.
2. **Programador de Mecánicas:** Es el encargado de convertir el diseño de juego en código ejecutable. Entre sus tareas se encuentra definir las físicas del mundo del juego, definir las funciones de los distintos objetos y modelar el comportamiento de los personajes.
3. **Programador de gráficos 3D:** Es el responsable de implementar los sistemas para la creación y renderizado de gráficos 3D. El programador de gráficos 3D necesita contar con conocimientos avanzados en cálculo, matemática vectorial y matricial, trigonometría y álgebra.
4. **Programador de Interfaces:** Es el encargado de implementar los sistemas de interacción entre el jugador y el juego, normalmente interfaces de control, menús

1. ESTADO DEL ARTE

- y HUDs (Head-Up Displays).
5. **Programador de Audio:** Es la persona responsable de la implementación de los distintos sistemas que se utilizarán para reproducir música y sonido en el juego
 6. **Programador de Herramientas:** El programador de herramientas tiene la responsabilidad de crear las distintas herramientas que el resto del equipo pueda necesitar para realizar, o acelerar, su trabajo. Un tipo especializado de programador de herramientas es el programador del editor de niveles, debido a la importancia de esta herramienta para el desarrollo del juego y por la posibilidad de que dicho editor sea lanzado al público como parte del juego.
 7. **Programador de Red:** Es el encargado de escribir el código que permite a los juegos ser ejecutados entre varios equipos, ya sea código máquina de bajo nivel o la integración de un librería de alto nivel.



Figura 1.17: Jonh Romero es el co-fundador de ID Software y programador de juegos como Doom o Quake

Arte

Se denomina artista a la persona o grupo encargado de generar los componentes gráficos del juego: modelos 3D de personajes y objetos, texturas, diseño de menús e interfaces, bocetos, animaciones y demás.

Existen varias categorías de artistas distintas dependiendo de en qué rama se especialicen y de cuál sea su rol en la estructura del proyecto[Bet03]:

- **Director Artístico:** Asignado al artista con mayor experiencia en la industria, el papel del director artístico es el de organizar y coordinar al resto de artistas para que realicen correctamente su trabajo y el de revisar las piezas producidas para asegurarse de que son consistentes con el estilo artístico establecido.
- **Artista Conceptual:** El artista conceptual es el encargado de producir bocetos

provisionales que servirán como base para construir los gráficos definitivos del juego.

- **Artista 2D:** Los artistas 2D son expertos en las técnicas tradicionales del dibujo y pintura. Su rol es más notable en los juegos 2D, donde deben producir la mayoría de los componentes gráficos como fondos, *tiles* y *sprites*; pero también tienen un papel notable en los juegos 3D, donde suelen ser los encargados de diseñar interfaces gráficas, crear las texturas de los modelos 3D e incluso realizar trabajos ajenos al propio juego como la creación de imágenes promocionales.
- **Modelador 3D:** Es el encargado de producir los modelos 3D de los distintos componentes del juego tales como personajes, objetos, mapas... Es relativamente común que los modeladores 3D tengan ciertos conocimientos de programación debido a que eran necesarios para trabajar con los primeros programas de modelado 3d.
- **Animador:** Es el encargado de animar los diferentes elementos del juego, desde el simple movimiento de un molino de viento hasta las complicadas expresiones de una cara. Existen dos alternativas para realizar la animación: la técnica de keyframing, que consiste en realizar poses estáticas de los personajes que el programa utiliza para generar la animación; y la captura de movimiento, en la que los movimientos de un actor son capturados y transferidos al juego mediante un equipo especializado.
- **Storyboarder:** Es el artista encargado de diseñar escenas del juego. Para ello, el Storyboarder crea unas secuencias de arte conceptual que describen los tiempos, diálogos y eventos de las escenas, lo que permite valorarla y validarla antes de iniciar el costoso proceso de producción.



Figura 1.18: Akira Toriyama, el autor de Dragon Ball, ha trabajado como artista en juegos como Dragon Quest o Chrono Trigger

1. ESTADO DEL ARTE

Audio

El trabajo de audio en un videojuego viene en tres categorías: música, efectos de sonido y doblaje. Existen especialistas que se dedican exclusivamente a una sola de estas categorías, aunque no es raro encontrarse en pequeños estudios a una persona encargarse tanto de la música como del sonido. Reflejando los tipos de audio, los tres tipos de profesionales son[Bet03]:

1. **Músico:** Es el artista encargado de escribir las composiciones musicales que se escucharán a lo largo del juego. Es muy común que el músico también se haga cargo de interpretar sus composiciones mediante programas de síntesis de música, aunque las grandes producciones pueden permitirse contratar interpretaciones en vivo.
2. **Técnico de sonido:** Los técnicos de sonido son profesionales que se dedican a fabricar o adaptar sonidos para el proyecto en el que trabajen.
3. **Equipo de doblaje:** El trabajo de doblar un videojuego requiere del trabajo de varios profesionales. En primer lugar, está el actor de voz, un actor especializado que interpreta con su voz a uno o varios personajes del juego. El trabajo de los actores está supervisado por un director de doblaje, que además suele encargarse de adaptar el guion y de dirigir a los técnicos de sonido que van a grabar y manipular las voces.



Figura 1.19: Yoko Shinomura es una compositora conocida por su trabajo en videojuegos como Final Fantasy o Street Fighters

Testing

El Aseguramiento de la Calidad (o QA por sus siglas en inglés) es un requisito clave para el desarrollo de un videojuego, a la vez de un proceso lento y costoso que debe comenzarse lo más pronto posible para evitar un sobrecoste[Bet03]. El trabajo del tester es el de revisar las distintas versiones del juego en busca de fallos para que los desarrolladores puedan arreglarlos.

Normalmente, los testers de un juego se agrupan en equipos, dirigidos por un jefe de testing. Cada equipo de testers se encarga de revisar una faceta distinta del juego, el equipo principal se encarga de probar la jugabilidad y los modos de juego individuales, el equipo multijugador se ocupa de revisar la jugabilidad en línea, así como los componentes técnicos de las conexiones, el equipo de compatibilidad prueba el juego en diversas plataformas y PCs con distintos componentes y el equipo de localización comprueba que se halla realizado una correcta traducción a distintos idiomas.

Para evitar la pérdida de punto de vista critico que el equipo principal y el equipo de multijugador pueden sufrir tras haber trabajado con el juego desde el principio del desarrollo, es normal cambiar los equipos en las últimas etapas del desarrollo por equipos nuevos que no estén involucrados en el juego.

Junto al trabajo de los equipos profesionales de testing se suelen realizar campañas de Beta Testing. El Beta testing consiste en liberar una versión incompleta del juego para que jugadores aficionados los prueben. Las resultados y opiniones de los jugadores son recogidos para utilizarse en el desarrollo de la versión completa. Para realizar una exitosa campaña de beta testing es necesario contar con uno o más organizadores que puedan gestionar la retroalimentación de los usuarios.

Producción

La función principal del productor es servir de puente entre el equipo de desarrollo y el resto de la empresa. El productor debe tener un conocimiento profundo del juego y de los demás miembros del equipo, de forma que pueda explicarlo de forma correcta en las muchas reuniones que se tendrán con otros departamentos, como por ejemplo el de marketing [Bat04].

El productor se encarga de realizar la gestión del proyecto, coordinando al equipo, realizando la programación de las etapas del proyecto y gestionando los posibles riesgos.

Existen tres tipos de productores dependiendo de su especialidad. Estos son:

1. **Productor Externo:** Este tipo de productor trabaja para la compañía editora y se encarga de supervisar al equipo de desarrollo para asegurarse de que se cumplen los acuerdos establecidos por ambas partes.
2. **Productor Interno:** Esta clase de productor trabaja en la compañía desarrolladora y se encarga tanto de realizar una gestión interna del proyecto como de actuar de representante de del equipo.
3. **Asistente de Producción:** Los asistentes de producción se encargan de realizar las tareas a las que el productor jefe del proyecto no puede dedicarse personalmente. Normalmente se trata administrar detalles concretos como administrar

1. ESTADO DEL ARTE

recursos, realizar el papeleo o gestionar los servidores y la página web.

1.3 Motores de juegos

1.3.1 Descripción

Un motor de juego es un framework software que facilita el desarrollo de videojuegos proveyendo al programador de la funcionalidad general necesaria para cualquier juego[War].

El término “Motor de Juegos” se remonta a mediados de los años noventa, cuando apareció el género de los juegos de Tiros en Primera Persona. Doom, uno de los primeros juegos de este género, había sido diseñado de forma que existía una separación bien definida entre los componentes software principales (como el motor de renderizado 3D o el sistema de detección de colisiones) y los assets gráficos, los mundos y las reglas del juego. Esta separación permitía el desarrollo de nuevos juegos solo cambiando el arte, niveles o armas de títulos anteriores, manteniendo intacto el motor[Gre09].



Figura 1.20: Heretic (Raven Software, 1994), es un juego desarrollado con el motor de Doom.

Este concepto inició las comunidades de “Modding”, que son grupos de aficionados que desarrollaban juegos nuevos modificando juegos antiguos, y para finales de los noventa, juegos como Quake III y Unreal habían sido diseñados pensando en la reusabilidad de su código. Actualmente, la mayor parte de las compañías desarrolladoras de juegos adquieren la licencia de motores desarrollados por terceros, mucho más económico que desarrollar desde cero todos los componentes.

La linea que separa el motor del juego suele ser difusa y depende en gran medida del juego concreto. El principal factor que se utiliza para distinguir un motor de juego de un juego que haya sido desarrollado de forma “íntegra” es la presencia de una Arquitectura orientada a datos. Se trata de un paradigma de programación que se basa en diseñar software con la intención de procesar datos, en lugar de ejecutar secuencias instrucciones fijas.

Idealmente, debería ser capaz de “reproducir” cualquier juego a partir de los datos de su contenido, de la misma forma que un programa reproductor de música leer y reproducir canciones. Sin embargo, en la realidad los motores de juegos suelen estar optimizados para un determinado género juego o una plataforma específica debido a que de esa forma es posible obtener software más eficiente y con mayores prestaciones, aplicando técnicas y patrones de diseño específicos de esos géneros/plataformas.

Componentes de un Motor

Los motores de Juego son softwares muy complejos, por lo que suelen estar construidos a partir de módulos independientes, lo que facilita enormemente su mantenimiento. Normalmente, un motor de juegos se compone de los siguientes módulos[Gre09]:

- **El Nucleo:** Se trata de una aplicación compleja que recoge gran cantidad de herramientas y utilidades necesarias para el desarrollo del juego. El núcleo suele incluir una librería de funciones matemáticas, herramientas para la gestión eficiente de memoria, estructuras de datos y clases personalizadas, etc.
- **Motor de Renderizado:** Posiblemente el componente más grande y complejo del juego, el motor de renderizado es el software encargado de generar los gráficos del juego. Estos motores suelen estar construidos siguiendo una estructura de capas: primero el **render de bajo nivel**, que se encarga de dibujar primitivas a la mayor velocidad posible; el **grafo de escena** determina qué sección de la escena es visible, lo que permite reducir el número de llamadas al render de bajo nivel; la capa de **efectos visuales**, que contiene el sistema de partículas, los efectos de pantalla completa, o las luces dinámicas; y finalmente la capa de **frontend**, la cual sirve para el renderizado de imágenes 2D que se superpondrán a la escena tridimensional del juego, como los menús, el HUD (Head Up Display) o videos pre-renderizados. Aparte del motor gráfico, los motores de juego actuales suelen incluir también un **Sistema de Animación**, el cual permita dotar de movimientos naturales a los personajes y elementos del juego.
- **Gestor de Recursos:** Se trata de una interfaz que permite un acceso unificado a los distintos assets que forman el juego (modelos, texturas, sonidos, scripts...).
- **Motor de Físicas:** El motor de físicas permite realizar la detección de colisiones entre entidades del juego, así como la simulación de comportamientos físicos

1. ESTADO DEL ARTE

realistas para dichas entidades. Hoy en día, las compañías no suelen programar sus propios motores de físicas, en su lugar adquieren motores desarrollados por terceros, como *Havok*¹¹ o *PhysX*¹².

- **Entrada y Salida del Jugador:** Este sistema se encarga de gestionar la información de entrada del jugador, suministrada mediante el mando de juego o el teclado y ratón. El sistema toma la información en bruto de entrada y permite al programador acceder a ella de forma más útil, limpiando los datos de entrada, creando eventos de activación de teclas y proveyendo de sistemas para mapear funciones a distintas teclas o botones y para detectar secuencias de pulsaciones. Este sistema también provee funciones para la salida de datos relacionado con los mandos de control, como activar y desactivar la vibración o emitir sonidos.
- **Sistema de Audio:** Es el componente encargado de la reproducción de la música y efectos de sonido del juego. Aunque su complejidad depende en gran medida de las necesidades del motor concreto, la mayoría incluyen sistemas para producir efectos como sonido 3D o música dinámica.
- **Networking:** Son los sistemas encargados de realizar la conexión del juego con Internet para, en la mayoría de los casos, realizar partidas en línea con otros jugadores. El soporte de sistemas para el juego multijugador tiene un gran impacto en la mayoría de componentes del motor, por lo que estos suelen ser desarrollados pensando desde el principio en el modo multijugador, e implementando el modo de un jugador como un caso específico del multijugador.
- **Fundamentos de la Jugabilidad:** Esta capa implementa una serie de sistemas que permiten implementar la Jugabilidad. Suele incluir un lenguaje de Scripting, un sistema de eventos, inteligencia artificial, cámaras...
- **Herramientas de depuración:** Estas herramientas facilitan la tarea de depurar y optimizar el juego. Incluyen herramientas para dibujar en pantalla, consolas de comandos, sistemas para grabar y reproducir sesiones de juego...

1.3.2 Ejemplos de Motores

Unity 3D

Unity¹³, conocido popularmente como **Unity3D**, es un motor de juego con su propio Entorno de Desarrollo Integrado (IDE) lanzado en el año 2005 por la compañía Unity Technologies. El desarrollo de este motor comenzó en el año 2002, encabezado por los desarrolladores David Helgason, Joachim Ante y Nicholas Francis. Se trata de uno de los motores más utilizados del mercado, especialmente para el desarrollo de juegos para

¹¹<https://www.havok.com/physics/>

¹²<https://www.geforce.com/hardware/technology/physx#source=gss>

¹³<https://unity3d.com/unity>

dispositivos móviles (más del 30 % de los 1000 juegos más exitosos para dispositivos móviles fueron desarrollados con este motor).

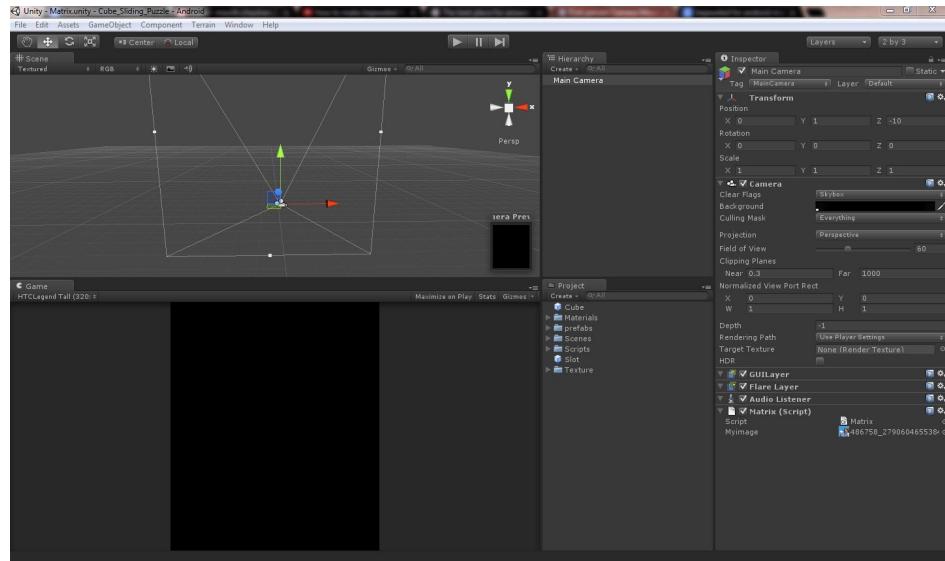


Figura 1.21: Captura del entorno de desarrollo de Unity

La cualidad más importante de Unity es la facilidad de su uso: El motor cuenta con una arquitectura de componentes fácil de entender, el proceso de exportación es muy simple y además cuenta con una tienda de Assets integrada. El Scripting en Unity se realiza en los lenguajes **C#** o **UnityScript** (una variante de JavaScript). Unity ofrece la posibilidad de exportar juegos a 27 plataformas distintas, desde dispositivos móviles a consolas de sobremesa, pasando por los sistemas operativos más comunes de los ordenadores personales. Debido a que su uso es muy extendido, Unity cuenta con una amplia base usuarios, por lo que es sencillo encontrar ayuda en los foros de desarrollo. La facilidad de uso viene a costa de la potencia: su motor gráfico no es tan potente como el de otros motores de la competencia y su rendimiento no es muy óptimo para el desarrollo de juegos de gran envergadura.

Unity Technologies ofrece distintos planes de pago a los desarrolladores interesados en adquirir la licencia de su motor: La versión gratuita, Unity Personal, que ofrece la funcionalidad básica; Unity Plus que por 35€ al mes ofrece mayor personalización y herramientas de monetización y análisis de rendimiento; y la versión profesional, Unity Pro (125€ al mes) la cual ofrece las prestaciones de la versión Plus más acceso al código fuente y mejor atención al cliente.

Unreal Engine

El **Unreal Engine**¹⁴ es un popular motor de juego desarrollado por la compañía Epic Games. Originalmente desarrollado como motor propietario para el juego Unreal

¹⁴<https://www.unrealengine.com/>

1. ESTADO DEL ARTE



Figura 1.22: A Hat in Time (Gears for Breakfast, 2017)



Figura 1.23: Hollow Knight (Team Cherry, 2017)

Figura 1.24: Juegos desarrollados con Unity3D

(1998), Epic Games pronto empezó a cerrar tratos con otras compañías que querían utilizar el motor en sus proyectos. Actualmente el motor se encuentra en su versión 4 y es uno de los más populares del sector, habiendo ganado incluso el premio Guinness al "motor de juegos más exitoso" con un total 408 juegos (a fecha de julio de 2014) desarrollados con Unreal.

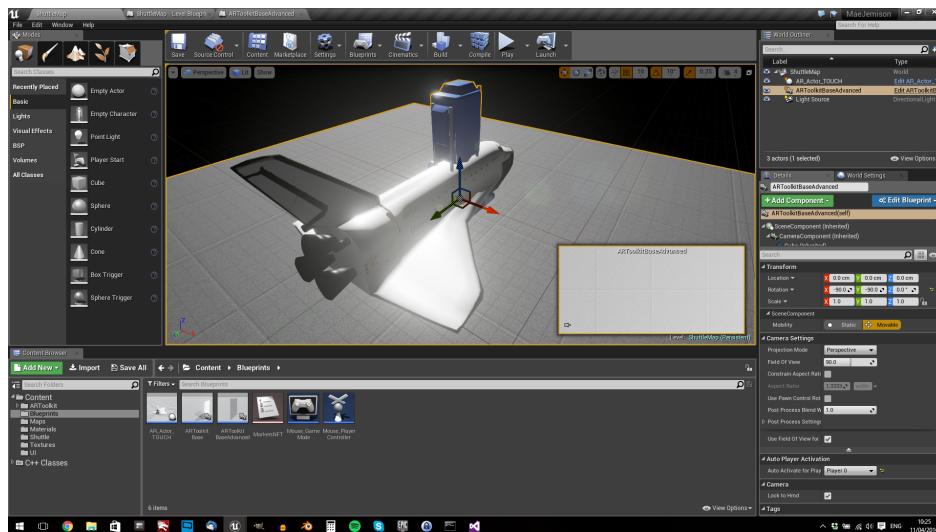


Figura 1.25: Captura del entorno de Unreal Engine

Unreal Engine es un motor orientado al desarrollo de juegos “AAA”, proyectos de gran envergadura llevados por equipos grandes. Uno de sus puntos fuertes es su potente motor de rendering el cual da soporte a gráficos fotorrealistas y permite el uso de efectos de post-procesados complejos entre otras características. El scripting en Unreal se realiza mediante el sistema **Blueprint** de Scripting visual, el cual permite programar conectando de forma gráfica bloques de código. El motor permite también escribir código directamente en C++, lo que aumenta su flexibilidad. El paquete de herramientas del motor también incluye programas como generadores de terreno, editores de mate-

riales, herramientas para animación... Sin embargo, se trata de un motor complicado y difícil de aprender a utilizar, además de que la potencia que requiere lo hace poco adecuado para el desarrollo para plataformas móviles.



Figura 1.26: Unreal (Epic Games, 1998)

Figura 1.27: Batman: Arkham Knight (Rocksteady Studios, 2015)

Figura 1.28: Juegos desarrollados con Unreal Engine

La licencia de uso de Unreal Engine es gratuita, sin embargo, en proyectos comerciales Epic Games cobra un 5 % de las ganancias a partir de los 3.000\$ por trimestre. En casos especiales, es posible negociar otros tipos de licencias con Epic Games.

Game Maker Studio

Game Maker Studio¹⁵ es un motor de juegos desarrollado por Yoyo Games. El programa fue lanzado originalmente en 1994 bajo el nombre de **Amino** como una herramienta para la creación de animaciones. Desde entonces, el programa ha ido evolucionado hasta convertirse en un motor de juegos de calidad profesional.

Game Maker Studio está diseñado para ser muy sencillo de usar: su principal uso es como una herramienta para gente sin conocimientos de programación, la cual permite desarrollar juegos sin necesidad de escribir una línea de código; o para el desarrollo rápido de juegos pequeños o prototipos. La programación de scripts en Game Maker se realiza mediante el sistema “**Drag and Drop**”, con el que se programa conectando diversos bloques de código; o el mediante un lenguaje de programación propio llamado **Game Maker Language**. El motor permite exportar con facilidad a distintas plataformas como PC, dispositivos móviles o HTML5. El entorno integrado de Game Maker incluye herramientas complementarias como un editor gráfico y un editor de mapas para centralizar el desarrollo. Sin embargo, la sencillez del motor también se refleja en su potencia: Game Maker Studio carece de soporte para gráficos tridimensionales complejos, y su arquitectura dificulta el desarrollo de proyectos de gran envergadura.

¹⁵<https://www.yoyogames.com/gamemaker>

1. ESTADO DEL ARTE

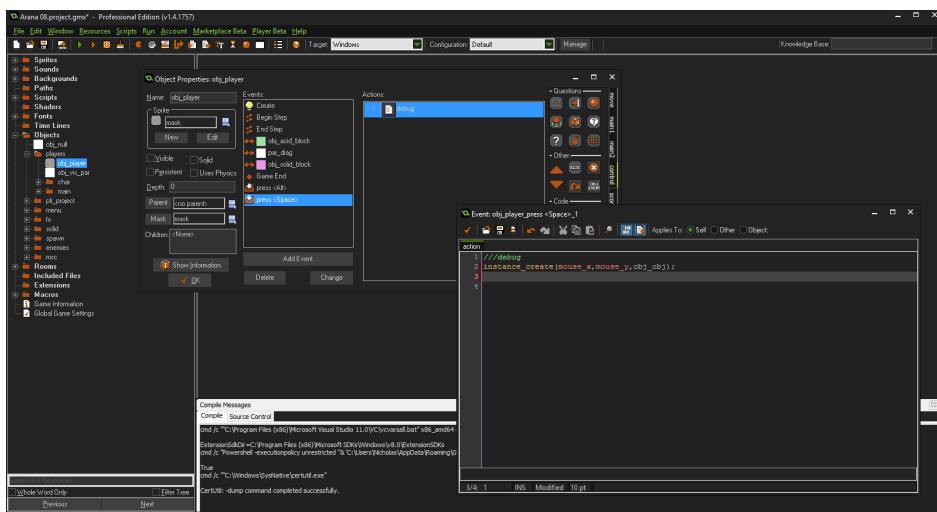


Figura 1.29: Captura del entorno de Game Maker Studio



Figura 1.30: Undertale (Toby Fox, 2015)

Figura 1.31: Hyper Light Drifter (Heart Machine, 2016)

Figura 1.32: Juegos desarrollados con Game Maker

La licencia de la versión actual de Game Maker Studio (Game Maker Studio 2) se encuentra a la venta por distintos precios dependiendo de la plataforma de distribución para la que se quiera trabajar: desde la versión básica por 39\$ anuales hasta la versión profesional permanente con posibilidad de exportación a IOS, Android y consolas por 399\$. Existen también una versión de prueba gratuita que cuenta con unas prestaciones reducidas y un plan de pago para su uso en centros educativos.

1.4 Inteligencia Artificial en Videojuegos

1.4.1 Historia

Desde los principios de la inteligencia artificial, los expertos han dedicado una importante cantidad de tiempo y esfuerzo para construir sistemas inteligentes con el propósito de que pudiesen jugar a juegos con un nivel igual o superior al humano. Este enfoque en el estudio de los juegos se debe a que ofrecen un campo de estudio ideal

para la inteligencia artificial: los juegos son problemas complejos que ofrecen desafíos para múltiples campos de la inteligencia artificial y además son tan populares que se dispone de cantidades inmensas de información sobre ellos[YT18].

Los primeros programas capaces de jugar contra humanos surgieron en los años cincuenta. Uno de los ejemplos más antiguos es el algoritmo ajedrecista de Alan Turing de 1948[Tur53], el cual era “ejecutado” en papel por un humano que seguía manualmente los pasos del algoritmo. El primer Software capaz de jugar a un juego fue la inteligencia artificial para el juego OXO, programada por Alexander Douglas en 1952. En 1959, Arthur L. Samuel[Sam59] programó una inteligencia artificial para jugar a las Damas la cual contaba con un sistema de aprendizaje.

Sin embargo, estos programas eran muy simples y no planteaban reto alguno contra jugadores experimentados cuando se trataba de juegos con una cierta complejidad, como el ajedrez. Hubo que esperar a los años noventa para que empezaran a surgir programas capaces de derrotar a grandes maestros de distintos juegos: en el año 1994 el programa Chinook Checkers derrotó al campeón mundial de la Damas Marion Tinsley¹⁶ y 3 años más tarde el super-ordenador Deep Blue venció a Gary Kasparov¹⁷. Hoy en día, la inteligencia artificial ha demostrado ser capaz de superar a los jugadores humanos en casi cualquier juego, con ejemplos tales como la inteligencia artificial Watson ganando el concurso de televisión Jeopardy en 2011¹⁸ o el programa AlphaGo derrotando a Ke Jie, el jugador número uno de Go¹⁹ (un juego con una complejidad varios ordenes de magnitud superior al ajedrez).

Hoy en día, con el auge de los videojuegos, ha comenzado el estudio para la resolución de juegos continuos en tiempo real (en contraste a los juegos de mesa tradicionales, que son discretos). Estos juegos presentan un desafío mayor, pero ya se han realizado avances, como el algoritmo desarrollado por Google DeepMind en 2014 para la resolución de varios juegos de la consola clásica Atari 2600²⁰.

1.4.2 IA Aplicada a Videojuegos: Contexto Actual

El uso de la Inteligencia Artificial en la industria del videojuego difiere en varios puntos a su aplicación habitual en el campo académico de los juegos que hemos visto en el apartado anterior. La principal diferencia es el tipo de problema que se intenta resolver utilizando inteligencia artificial en cada una de las ramas: en la Inteligencia Artificial aplicada a jugar a juegos se busca obtener un sistema capaz de jugar de manera óptima para derrotar a cualquier adversario humano, sin embargo, en la In-

¹⁶<https://webdocs.cs.ualberta.ca/~chinook/project/>

¹⁷<http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/>

¹⁸https://www.youtube.com/watch?v=WFR3lOm_xhE

¹⁹<https://www.theverge.com/2017/5/25/15689462/alphago-ke-jie-game-2-result-google-deepmind-china>

²⁰<https://deepmind.com/research/publications/playing-atari-deep-reinforcement-learning/>

1. ESTADO DEL ARTE



Figura 1.33: El Gran Maestro de ajedrez Gari Kaspárov (izquierda) enfrentándose al ordenador Deep Blue

teligencia Artificial orientada a videojuegos lo que se busca es crear sistemas con los que mejorar la experiencia de juego del jugador. Esta diferencia de objetivos hace que la Inteligencia Artificial no se aplique únicamente a la creación de oponentes virtuales, sino también al modelado del comportamiento de Personajes No Jugadores y a la Generación Procedimental de Contenido[YT18].

En el ámbito de los videojuegos, a la Inteligencia Artificial que interacciona con el juego como un jugador más suele llamarse "Bot". Estos bots predominan en juegos competitivos donde es necesario un oponente con un grado de inteligencia elevado para suponer un reto al jugador, como los juegos de estrategia, los juegos de lucha o los juegos de disparos en primera persona. Debido al elevado coste y complejidad de desarrollar una Inteligencia Artificial potente para estos juegos, por no hablar de la potencia requerida para ejecutarla, en la mayor parte de los videojuegos la Inteligencia Artificial "hace trampas", juega teniendo acceso una serie de ventajas que los jugadores humanos no tiene. Estos sistemas podrían, por ejemplo, acceder a información oculta del jugador (como la posición o recursos) a la hora de planificar estrategias o dispondrían de más y mejores recursos que sus adversarios.

En la mayoría de los juegos, la Inteligencia Artificial no se dedica al modelado de bots como los descritos anteriormente, sino que lo más común es que se utilice para controlar el comportamiento de Personajes no Jugadores, o NPCs (siglas inglesas de Non-Player Character). Estos pueden tener comportamientos muy variados dependiendo de su papel en el juego: pueden actuar como adversarios, servir de ayuda para el jugador, formar parte de un puzzle, contar una historia o simplemente formar parte del

trasfondo de la acción. En el diseño de NPCs, lo que se busca son dos cosas: crear una Ilusión de Inteligencia, de forma que los jugadores crean que el NPC es un ser inteligente, aunque sea controlado por un código muy sencillo, y, en especial con adversarios, que sea predecible, de forma que el jugador pueda desarrollar estrategias para enfrentarse/interaccionar con ellos.



Figura 1.34: En Los Sims (Maxis, 2000-2014), los personajes pueden tomar decisiones basándose en sus gustos y necesidades.

La otra aplicación principal en los videojuegos es la Generación Procedimental de Contenido. La generación Procedimental de Contenido es el nombre que reciben los métodos que permiten generar el contenido de un juego de forma automática o con solo un mínimo de intervención humana. Actualmente, la mayor parte de los juegos que hacen uso de la generación procedural la utilizan para la creación automática de mapas o niveles o para la creación de objetos. La generación procedural de contenido puede utilizarse tanto como una herramienta durante el desarrollo, que serviría para generar contenido que luego sería refinado por los desarrolladores; o podría formar parte del juego final, generando nuevo contenido al gusto del jugador de forma automática.

1.4.3 Métodos de la IA

Existen varios tipos de métodos y algoritmos los cuales pueden ser utilizados para construir inteligencias Artificiales. La elección entre los distintos tipos de métodos debe realizarse dependiendo del tipo de problema que se intenta resolver y de los recursos de los que se dispone, tanto las prestaciones del dispositivo en el que van a ser implementados como margen de tiempo máximo de la ejecución del algoritmo[YT18].

1. ESTADO DEL ARTE



Figura 1.35: Minecraft (Mojam, 2009) es un ejemplo claro de generación procedural de terrenos.

Los Métodos de la inteligencia artificial pueden ser agrupados en las siguientes categorías, debido a sus características y aplicaciones similares:

Métodos Ad Hoc

Esta clase de métodos de IA es una de la primera y, en el sector del videojuego, la más común. Su nombre proviene de la locución latina que, traducida, significa “para esto”, y hace referencia a que se tratan de soluciones precisas para problemas concretos, las cuales no pueden generalizarse ni aplicarse a otros problemas distintos[YT18].

Pese al notable problema que provoca la falta de reusabilidad de estos métodos, son los más utilizados en el desarrollo de videojuegos. Esto se debe a que, por lo general, son muy fáciles de diseñar, visualizar, implementar y depurar; además requerir de muy pocos recursos cuando se aplican a problemas pequeños.

Algunos tipos de métodos Ad hoc son:

- **Máquinas de estados finitos:** Es un tipo de sistema experto en el que la información es representada por un grafo dirigido, donde los nodos representan **Estados** en los que se puede encontrar un comportamiento, proceso o algoritmo y las aristas las **Transiciones** condicionales entre los distintos estados. Fácil de diseñar e implementar, pero su complejidad crece muy rápido, por lo que no es apto para problemas grandes.
- **Arboles de comportamiento:** Este tipo de sistema experto utiliza una estructura en árbol para el modelado de información, donde cada nodo es un **comportamiento**. El algoritmo recorre el árbol en profundidad, ejecutando el comportamiento de dicho nodo. El comportamiento del nodo padre depende del resultado de los hijos. Los arboles de comportamiento ofrecen una mayor flexibilidad que las máquinas de estados a cambio de mayor complejidad.

- **IA basada en utilidad:** Esta técnica se basa en el uso de una **Función de Utilidad**, la cual asigna un valor de utilidad a todas las opciones del agente inteligente basándose en información del entorno. El agente elige la opción que tenga una utilidad más alta.

Búsquedas en Árbol

Una de las bases de la inteligencia artificial son los **algoritmos de búsqueda en árbol** son una de las bases de la inteligencia artificial, dado que la mayor parte de los problemas de la inteligencia artificial podrían plantearse usando este tipo de algoritmos[YT18].

La búsqueda en árbol consiste en la construcción de un **Árbol de búsqueda**, un tipo de grafo dirigido en el cual los nodos o hojas representan estados mientras que las aristas o ramas representan las acciones que provocan la transición de un estado a otro. El agente inteligente recorre el árbol partiendo del nodo raíz buscando la secuencia de ramas que resuelvan el problema siguiendo un algoritmo concreto. Cuando se aplican a juegos, los arboles de búsqueda suelen aplicarse para modelar el estado del juego: la inteligencia artificial recorre el árbol buscando la secuencia de acciones que lleve al estado de victoria, o evitando el estado de derrota.

Existen numerosos algoritmos de búsqueda distintos, los cuales pueden agruparse en las siguientes categorías:

- **Búsqueda no informada:** Se trata de los algoritmos más básicos, en los que se realiza la búsqueda sin ningún tipo de información adicional sobre el objetivo. Sus variantes más simples son el algoritmo **Primero en Anchura**(explora todas las acciones de un estado antes de pasar al siguiente) y el **Primero en Profundidad**(explora una secuencia de ramas todo lo que puede antes de volver e intentar otra secuencia distinta)
- **Búsqueda Primero el Mejor:** En estos algoritmos se cuenta con información adicional sobre el nodo objetivo, la cual se utiliza para determinar que nodos deben explorarse primero. Un tipo de algoritmos Primero el Mejor son los algoritmos **A***, en los cuales los nodos se seleccionan basándose en su **Coste**(suma de la distancia entre el nodo y el nodo raíz y la distancia estimada al objetivo).
- **Búsqueda MiniMax:** Este tipo de algoritmos se utilizan para resolver problemas que involucran a dos adversarios enfrentados. En estos algoritmos se va alternando entre jugadores **Min** y **Max**, los cuales intentan llegar a sus respectivos estados de victoria opuestos. El espacio de búsqueda de estos algoritmos suele ser muy grande, por lo que suelen usar funciones de evaluación para evitar recorrer el árbol de búsqueda completo.

1. ESTADO DEL ARTE

- **Árbol de búsqueda de Monte Carlo:** Se trata de una familia de algoritmos diseñados para resolver problemas **No deterministas** y/o de **Información Imperfecta**. Para evaluar la calidad de un estado dado, el algoritmo utiliza simulaciones aleatorias de partidas a partir de ese estado. La siguiente acción será con la que empezó el mayor número de partidas victoriosas.

Algoritmos de Optimización

Los algoritmos de optimización, a diferencia de la búsqueda en árbol, se centran en obtener una solución correcta, ignorando los pasos que llevaron a esta. Para ello, el algoritmo empieza tomando una solución sub-optima, la cual va modificando en múltiples iteraciones utilizando una **función de Aptitud** como guía hasta obtener una solución con una Aptitud máxima[YT18].

Existen varios tipos de algoritmos de optimización, dependiendo de los métodos que elijan para la selección de la solución inicial, para la evaluación de aptitud o para la modificación:

- **Búsqueda Local:** Este tipo de algoritmos consisten en, dada una solución, revisar todas las soluciones que difieran de dichas soluciones por una distancia mínima. Si alguna de las soluciones mejora a la solución original, se remplaza la solución original por la nueva solución y repite el algoritmo; si no, la solución original es elegida como la solución óptima.
- **Algoritmos Evolutivos:** Los algoritmos Evolutivos son un tipo de algoritmos de optimización basados en la evolución por selección natural Darwiniana que se observa en la naturaleza. Su funcionamiento se basa en un conjunto amplio de soluciones candidatas. EN cada iteración del algoritmo, las distintas soluciones se “cruzan” para obtener soluciones mixtas, las cuales son evaluadas. Finalmente, se forma un nuevo conjunto de soluciones a partir de las soluciones más exitosas. El algoritmo se repite hasta obtener la solución más óptima.

Aprendizaje Automático

El aprendizaje automático es una rama de la computación que permite dotar a los ordenadores de la capacidad de “aprender” a realizar una tarea utilizando datos en lugar de programarlo específicamente para esa tarea[Sam59]. Los algoritmos de aprendizaje automático suelen aplicarse a problemas donde es muy difícil, o incluso imposible, programar un algoritmo implícito que pueda resolverlo, como por ejemplo el filtrado de correos o la visión por ordenador.

Una forma de clasificar los algoritmos de aprendizaje automático es basándose en el tipo de **Retroalimentación** que reciben. De esta forma, surgen las siguientes categorías:

- **Aprendizaje Supervisado:** Se trata de algoritmos que extraer los atributos y características comunes de los integrantes de grupos etiquetados. Estos algoritmos funcionan recibiendo conjuntos de muestra formado por datos etiquetados en distintos grupos y categorías. Analizando los conjuntos de muestra, el algoritmo debe ser capaz de asignar nuevos datos a la categoría correcta. Existen muchos tipos de algoritmos de Aprendizaje supervisado, como por ejemplo las **Redes Neuronales**, que funcionan imitando la estructura de las neuronas del cerebro.
- **Aprendizaje por Refuerzo:** Los algoritmos de Aprendizaje por Refuerzo se inspiran en el **Conductismo Psicológico**, basándose en la forma en la que los animales aprenden a tomar decisiones basándose en los estímulos positivos y negativos de su entorno. Estos algoritmos suelen implementarse mediante un agente inteligente que interacciona con un entorno mediante acciones. Con cada acción, el agente recibe un estímulo positivo o negativo, que almacena con la intención de descubrir la secuencia de acciones que maximice el estímulo positivo a largo plazo mediante técnicas similares a los algoritmos de Optimización
- **Aprendizaje No Supervisado:** El aprendizaje No Supervisado son un conjunto de algoritmos que sirven para encontrar asociaciones y patrones en un conjunto de datos sin ningún tipo de información de refuerzo adicional. Existen varias aproximaciones a este tipo de algoritmos, como el **Clustering**, que consiste en agrupar elementos de un conjunto basándose en su similitud entre ellos y su diferencia con el resto de grupos.

1.4.4 Futuros campos de aplicación

La Inteligencia Artificial orientada a juegos evoluciona de manera paralela a los propios videojuegos, que viven ahora más que nunca una época dorada debido al aumento constante tanto en popularidad y prestigio, lo que supone una mejora tanto en la potencia de las máquinas que los ejecutan como en las técnicas que se utilizan en su desarrollo. Esta situación cambiante ha causado la aparición de nuevos problemas que se esperan poder solucionar con el uso de técnicas de inteligencia artificial.[YT18] A continuación mencionaremos algunos de los futuros campos de aplicación de las técnicas de inteligencia artificial:

Una las posibles aplicaciones de la Inteligencia Artificial es la de realizar **Testing** de juegos. El programa jugaría a los juegos en busca de fallos tanto informáticos como de diseño (como problemas de balanceo o maneras de hacer trampas en el juego) de forma automatizada, ahorrando una gran cantidad de trabajo a los desarrolladores. Aunque ya existen herramientas que ofrecen una forma rudimentaria de testing automático, aun es necesario mejorar aspectos como la categorización de los errores encontrados.

La **Minería de Datos de Juego** es otro uso prometedor de la Inteligencia Artificial.

1. ESTADO DEL ARTE

Se trata de una técnica alternativa al testing habitual de juegos en la que se recolecta y analiza la información de comportamiento de la base de jugadores de un juego dado con el fin de mejorar dicho juego [Yan12]. Esta técnica se ha popularizado gracias a la proliferación de juegos con un fuerte componente online que facilita la recogida de datos. Sin embargo los volúmenes de datos con los que se trabaja son tan masivos que los algoritmos de minería de datos actuales no son capaces de analizarlos completamente [Yan12].

La **Dirección de Juegos** consiste en una Inteligencia Artificial que modifica eventos del juego en tiempo real basándose en las reacciones de los jugadores con acciones tales como modificar la dificultad, reproducir música y sonido o modificar el entorno con tal de mejorar la experiencia del jugador. Actualmente, los juegos que mejor ha implementado un sistema con esta propiedad es la saga Left 4 Death de Valve²¹. Se trata de una rama con mucho potencial que aún no ha sido explorado completamente.

Finalmente, una tarea de gran importancia para los juegos online, a pesar de no formar parte de los juegos en si, es la **motorización de los chats**. El gran volumen de mensajes que se envían a través de los chats de los juegos online más populares hace que sea imposible la moderación manual, lo que crea un entorno de juego tóxico para los jugadores. Compañías como Riot (creadora del popular MOBA League of Legends) están empezando a utilizar algoritmos de aprendizaje automático para entrenar sistemas para detectar y eliminar los mensajes inapropiados de los chats²².



Figura 1.36: Ejemplo de comportamiento tóxico en League of Legends(Riot Games, 2009).

²¹<http://www.l4d.com/blog/>

²²<https://www.nature.com/news/can-a-video-game-company-tame-toxic-behaviour-1.19647>

ANEXOS

Referencias

- [Bat04] Bob Bates. *Game Design*. Thomson Course Technology, Boston, 2004.
- [Bet03] Erik Bethke. *Game Development and Production*. Wordware Publishing, Plano, Texas, 2003.
- [DEV17] Desarrollo Español del Videojuego DEV, editor. *Libro Blanco del Desarrollo Español del Videojuego*. Madrid, España, 2017.
- [DV15] David Villa David Vallejo, Carlos González. *Desarrollo de Videojuegos: Un Enfoque Práctico*. Ciudad Real, España, 2015.
- [Gre09] Jason Gregory. *Game Engine Architecture*. A K Peters, Ltd., Wellesley, Massachusetts, 2009.
- [Lyd] Bill Lydon. Industry 4.0 - only one-tenth of germany's high-tech strategy. URL <https://www.automation.com/automation-news/article/industry-40-only-one-tenth-of-germanys-high-tech-strategy>.
- [Sam59] Arthur L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, pages 210 – 229, 1959.
- [SKK14] James D. McCalley Siddhartha Kumar Khaitan. Design techniques and applications of cyber physical systems: A survey. *IEEE Systems Journal*, 9(2): 350 – 365, 2014.
- [Tur53] Alan M. Turing. Digital computers applied to games. *Faster than thought*, page 101, 1953.
- [War] Jeff Ward. What is a game engine? URL https://www.gamecareerguide.com/features/529/what_is_a_game_.php.
- [Yan12] Georgios N. Yannakakis. Game ai revisited. *Proceedings of the 9th conference on Computing Frontiers*, pages 285 – 292, 2012.
- [YT18] Georgios N. Yannakakis and Julian Togelius. *Artificial Intelligence and Games*. Springer, 2018. <http://gameaibook.org>.

Este documento fue editado y tipografiado con L^AT_EX empleando
la clase **esi-tfg** (versión 0.20180314) que se puede encontrar en:
https://bitbucket.org/arco_group/esi-tfg

[respeta esta atribución al autor]

