9° Congreso Nacional

CoNaIISI 2021

de Ingeniería Informática y Sistemas de Información

4 5 NOVIEMBRE

confedi
Consejo Federal de Decanos de Ingeniería

UTN FACULTAD REGIONAL MENDOZA
UNIVERSIDAD TECNOLÓGICA NACIONAL

Departamento de Ingeniería en Sistemas de Información

Red de Ingenierías en Informática / Sistemas de Información
RIISIC
ConFeDI

9° Congreso Nacional de Ingeniería Informática/Sistemas de Información

CoNaIISI

4 y 5 de noviembre de 2021

Universidad Tecnológica Nacional

Facultad Regional Mendoza

Memoria de Trabajos

9° Congreso Nacional de Ingeniería Informática/Sistemas de Información
9° CoNaIISI: Memorias de trabajos / Compilación de Marcela Fernandez; Matilde Césari; María Gabriela Martínez. - 1a ed. –
Ciudad Autónoma de Buenos Aires: Universidad Tecnológica Nacional, 2022.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-950-42-0213-4

1. Sistemas de Información. 2. Ingeniería Informática. I. Fernandez, Marcela, comp. II. Césari, Matilde, comp. III. Martínez, María Gabriela, comp. IV. Título
CDD 004.0711

Compilación de: Fernandez, Marcela
César, Matilde
Martínez, María Gabriela

Revisado por: Carbonari, Daniela
Caymes Scutari, Paola
Bianchini, Germán

ISBN 978-950-42-0213-4

Auspiciantes:

Estudio de Motores y Diseñadores en Videojuegos para entornos en 3D

Christian Parkinson, Roxana Martínez, Pablo Vera, Rocío Rodríguez
Centro de Altos Estudios en Tecnología Informática (CAETI). Universidad Abierta Interamericana, Ciudad Autónoma de Buenos Aires, Argentina
{Christian.Parkinson; Roxana.Martinez; PabloMartin.Vera; RocioAndrea.Rodriguez} @uai.edu.ar

Resumen

Este trabajo se enfoca en la importancia de los videojuegos en la actualidad, y su transformación desde la etapa inicial de su desarrollo hasta su implementación. Para ello, se presenta un estudio de los motores y diseñadores para entornos 3D más utilizados e influyentes en el mercado, por lo que se muestran sus características más sobresalientes. Para el caso del análisis de herramientas de Motores, se explican los aspectos más importantes para tener en cuenta desde su instalación, aspectos técnicos (memoria, disco, etc.), cómo así también su usabilidad para con la herramienta junto a las tendencias de consulta y uso. Desde el enfoque de los Diseñadores, este trabajo presenta una comparativa de las características más relevantes de éstos (si es Open Source, su Lenguaje de desarrollo, Plataformas utilizadas, Extensiones con las que trabaja, entre otras) a fin de brindar una guía que permita seleccionar la herramienta correcta. Finalmente se presentan algunas de las falencias encontradas en la utilización de estas herramientas, acompañadas de buenas prácticas y consejos a la hora de implementarlas en un proyecto de Videojuegos.

Palabras claves: *Videojuegos, Motores, Diseñadores,* GDD.

1. Introducción

La cultura de los videojuegos cada vez se encuentra más en auge. Distintas personas de todas las edades utilizan diversas herramientas de videojuegos. Este tipo de "medio nos abre a experiencias que difícilmente podrían vivirse en la vida cotidiana. Los videojuegos son un ejemplo claro de esto, los contenidos de los medios de comunicación y sus lenguajes pueden convertirse en instrumentos del pensamiento" [1]. Existen diversas definiciones de videojuegos entre ella se destaca: "El videojuego se podría definir como un hiperlenguaje dinámico-proyectivo, es decir, un instrumento que incluye diversos tipos de lenguaje distintos, como son el visual, el sonoro, el literario, gestual... todos ellos encuadrados en un mundo cambiante y dúctil a elección del creador del mismo y de los usuarios, de ahí la parte proyectiva, pues el usuario/a es quien verdaderamente encamina el juego como quiere jugarlo, proyecta su propia personalidad o una personalidad totalmente distinta con matices vinculados con su yo real o totalmente inversos, creando un alter ego irreconocible en el mundo real pero posible en el mundo virtual" [2]. Desde un enfoque más orientado a la pedagogía "los videojuegos fomentan la experiencia y el aprendizaje constructivista, que sostiene que el conocimiento es construido por los alumnos y que la discusión anima al debate y a la colaboración entre los estudiantes. Junto a la inmersión y otros atributos, permiten incluso en el caso educativo, proporcionar diversos contenidos de aprendizaje" [3]. Básicamente, "los videojuegos son los juguetes de la era tecnológica y el entretenimiento del futuro. Representan una de las entradas más directas a la cultura informática, a la cultura de la simulación, al mundo virtual, dotando a los niños de competencias necesarias para vivir en la sociedad digital" [4]. Con la integración de este tipo de aplicaciones en los sistemas operativos y en los teléfonos móviles, es decir, prácticamente en todos los dispositivos móviles, los videojuegos tienen una ubicuidad y una accesibilidad que no ha tenido ningún otro medio de la historia [5].

1.1. ¿Qué áreas intervienen en un Videojuego?

Para el desarrollo de un videojuego existen varios aspectos para tener en cuenta, en la **Figura 1**, extraída de [6], se muestran las áreas que intervienen en un proceso de desarrollo de videojuegos. Se puede observar que se visualizan distintas fases de trabajo, lo que requiere contar con varios perfiles profesionales al respecto. Estas fases involucran aspectos de programación, de diseño de imagen, de sonido, de animación, y de producción, entre otros. Por ejemplo, en la fase de diseño, se trabaja en tareas para el contexto de la historia, el guion, la jugabilidad y las reglas. Para el aspecto de Gráficos, se enfoca en las interfaces, modelos a utilizar, siendo éstos: 2D o 3D y las animaciones. Por otra parte, la fase de la programación posee la combinación de funcionalidades, gráficos, audios y reglas. A continuación, se presentan los factores de audio, teniendo en cuenta los efectos del sonido, música de fondo, diálogos

y demás. Finalmente, en la etapa de publicación, se orienta a la distribución y marketing del juego.

Figura 1. Áreas relacionadas en el desarrollo de un videojuego. Fuente [6].

Existe un documento importante en el contexto de los videojuegos donde todas las áreas recién expuestas documentan el proceso de desarrollo conocido como GDD (Game Design Document) [7], donde básicamente, una vez aclarado el juego a realizar, el siguiente paso del diseñador es plasmar todas las ideas a modo de redacción en un documento donde se explique detalladamente todas las características del juego. En este documento GDD, se va actualizando conforme el desarrollo avanza, ya que en cada iteración se irán modificando elementos del juego [8].

En la **Tabla 1**, se muestran algunos elementos que incluye un GDD, extraído de [7] y [9].

Tabla 1. Elementos que incluye un GDD.

Elemento	Descripción
Género	Clasificación según su naturaleza
Jugadores	Modalidad de juego (individual o colectivo). Si es multijugador, si éstos son humanos o entra la máquina.
Historia	Resumen de la historia del juego, de qué trata y cómo lo trata.
Look and Feel	A partir de los bocetos se define el aspecto gráfico y artístico del juego, colores, temas dominantes, musicalidad, técnicas de diseño 3D o 2D, posiciones de cámaras, etc.
Interfaz de Usuario	Se define como la manera en la que interactuará el jugador con el juego y con qué mecanismos contará para ello.
Objetivos	Cuáles son las metas del juego, de acuerdo con la historia de éste.
Reglas	Se define que se puede hacer y cómo se realizarán.
Características	Se especifican las principales características de cada personaje del juego y de los elementos que intervienen en éste.
Diseño de Niveles	Se describen qué niveles, según la historia o dificultad, tenemos, cómo serán éstos, cuántos serán, y qué dificultad y retos se plantean en cada uno de ellos.
Requerimientos técnicos	Se establecen los requerimientos técnicos de equipo que necesitará el juego para poder ejecutarse.
Marketing	Un mal estudio de marketing puede tirar por tierra la inversión millonaria de un producto de este tipo, es fundamental el trazado de las líneas de publicidad para nuestro juego.
Presupuesto	Se fija el presupuesto aproximado que se necesita para llevar a buen fin nuestro juego.

Todos los elementos que se muestran en la Tabla 1, son fundamentales a la hora de tener presente el desarrollo y diseño del videojuego. Cabe aclarar, que cada aspecto se debe enfocar en el alcance del proyecto del videojuego, como así también en las tareas planificadas para éste.

1.2. Géneros y Plataformas

En lo que respecta a los géneros de los videojuegos, algunos autores definen que, "el videojuego no es sólo un objeto de estudio «polifacético» respecto a las perspectivas de estudio que lo abordan, sino también por la gran diversidad de géneros de videojuego que se han ido consolidando con el paso de los años, tanto en el mercado como en la «cultura videolúdica»" [10].

Otros autores, sostienen que "cada plataforma posee unas características e interfaces que determinan en cierta medida tanto el género como el uso de los videojuegos. En los diferentes géneros se han popularizado los Arcade, cómo paddles (Pong o Breakout), laberintos (Pacman), shoot'em'up (disparar y olvidar como Space Invaders), simuladores, FPS (Juegos en primera y tercera persona, como DOOM), juegos deportivos, de lucha (Street Fighter), Puzzles (tetris), de estrategias y aventuras como Age of Empires o Adventure, hasta llegar a los mundos virtuales como los MMO (videojuego multijugador masivo en línea); pero en la evolución del hardware esta industria siempre ha estado de la mano de las consolas de videojuegos, es un sistema electrónico de entretenimiento para el hogar que ejecuta juegos electrónicos (videojuegos) que están contenidos en cartuchos, discos ópticos, discos magnéticos o tarjetas de memoria" [11]. Dada la gran diversidad de videos juegos se publicó en trabajos académicos 42 categorías [12] [13], las cuales luego fueron reducidas por otros autores a 11 categorías [14]. A continuación, se muestran dichas categorías extraídas de [15]:

- Acción – Aventura. El jugador es el protagonista de una historia y debe seguir un orden para progresar resolviendo determinadas situaciones y puzzles. Dichos puzzles suelen envolver acciones como manipular elementos del entorno o interactuar con otros personajes de la aventura.
- Lucha. Consisten en la recreación de combates, por lo general los combates se ven desde una perspectiva lateral o en tercera persona.

- Shooter (juegos de disparo en primera persona). El principal objetivo es disparar y matar enemigos, generalmente con armas de fuego aunque algunos videojuegos shooter requieren un nivel más alto de interacción y planificación para resolver determinadas situaciones.
- Plataforma. El jugador controla a un personaje que debe avanzar por el escenario evitando obstáculos físicos, ya sea saltando, escalando o agachándose.
- Puzzle. Son juegos que ponen a prueba la inteligencia del jugador para la resolución de problemas, que pueden ser de índole matemático, espacial o lógico.
- Racing. También conocidos como videojuegos de conducción, sitúan al jugador en un recorrido en el que debe llegar a una meta antes que sus contrincantes o dentro de un tiempo límite.
- Juegos de Rol (Role Playing Games también conocidos como RPGs). Estos videojuegos se caracterizan por una historia profunda y una evolución del personaje a medida que la historia avanza.
- Simuladores. Género que trata de representar situaciones de la vida real de la manera más fiel posible, a menudo sin un objetivo final definido, sino como mera recreación de una experiencia realista.
- eSports. Los videojuegos de deportes son aquellos que emulan deportes del mundo real. Entre ellos encontramos golf, tenis, fútbol, hockey, juegos olímpicos, etc.
- Strategy Games (Turn Based + Real Time). Se caracterizan por la necesidad de manipular a un numeroso grupo de personajes, objetos o datos, haciendo uso de la inteligencia y la planificación, para lograr los objetivos.
- Survival. Juegos en los que el personaje principal debe conseguir recursos suficientes para sobrevivir, por lo general en un entorno hostil o complejo. Están basados en la capacidad del jugador por encontrar combinaciones a los elementos de su entorno que le permitan construir herramientas o producir recursos para su subsistencia.

1.3. Ciclo de ejecución de un Videojuego

A diferencia de los programas de software tradicionales el ciclo de ejecución es completamente diferente. Una aplicación tradicional se encuentra en permanente estado ocioso, esperando que el usuario ingrese algún tipo de instrucción para que ejecute algún tipo de proceso. Los videojuegos tienen un ciclo de ejecución completamente diferente, el mismo consiste en un ciclo de actualización y dibujado permanente. (ver **Figura 2.**)

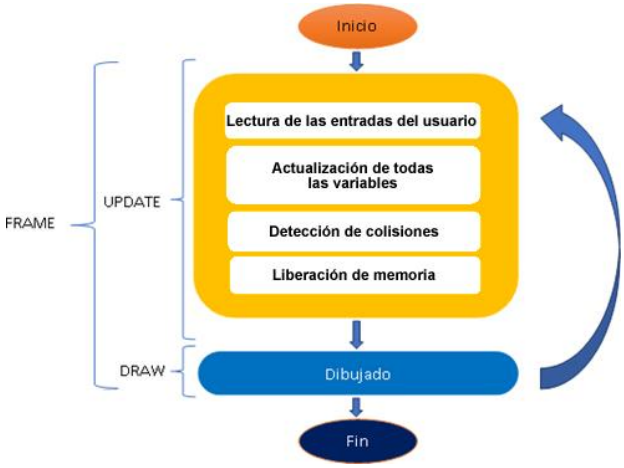

Figura 2. Ciclo de ejecución de un videojuego.

El proceso de actualización presenta cuatro pasos fundamentales:

1. **Lectura de las entradas del usuario:** se encarga de sensar todos los dispositivos de entrada, como ser el teclado, mouse, joystick, toques (touches) en caso de los dispositivos móviles, etc. En caso de recibir algún tipo de ingreso actualiza el estado de los sprites (elemento que representa un personaje u objeto dentro del juego) o los modelos relacionados con el jugador.
2. **Actualización de todas las variables:** en este proceso todos los elementos del juego se actualizan, por ejemplo, los movimientos de los enemigos, fondos, inteligencia artificial, etc. Esta actualización de los estados se produce independientemente que el usuario haya realizado ingreso alguno.
3. **Detección de colisiones:** En esta instancia con todos los elementos del juego actualizados se detectan las colisiones producto de los nuevos valores.
4. **Liberación de memoria:** Una vez actualizados todos los estados de todos los objetos, se eliminan todos los elementos que no se van a volver a utilizar dentro del juego. Por ejemplo, en un juego de naves la cantidad de disparos que se efectúan es muy numerosa, cuando un disparo colisiona con algún elemento o bien, cuando sale de la pantalla siguiendo una ruta hacia el infinito, ese elemento debe ser eliminado para liberar la memoria.

Una vez que se cumplen los cuatro pasos del proceso de actualización, el ciclo ejecuta el siguiente paso que es la representación de todos los elementos en pantalla conocida como “Render”. Dentro de los juegos 3D el objetivo es buscar una apariencia realista.

En este proceso se realizan todos los cálculos necesarios para representar la escena, en ambientes, en el caso de los escenarios 3D se aplican diferentes algoritmos para el manejo de las texturas, rigurosidad, volumen, colores, como también cálculos físicos sobre los materiales ante el efecto

de luces, como los reflejos, reflexión, refracción, oclusión, etc.

1.4 Problemática Detectada

A la hora de desarrollar un videojuego la cantidad de elementos a tener en consideración es muy numerosa, desde el punto de vista de diseño, programación, musicalización etc., para ello, existe una gran cantidad de herramientas, desde motores gráficos, diseñadores 2D y 3D, editores de sonido etc.

Dependiendo de las plataformas de destino, envergadura del desarrollo a realizar y el presupuesto necesario de inversión a la hora de adquirir licencias, se vuelve sumamente dificultoso poder seleccionar las herramientas para llevar adelante el trabajo. Por lo cual, en el presente trabajo se van a exponer y comparar diferentes criterios de selección entre los motores gráficos y diseñadores 3D más populares del mercado.

2. Relevamiento de Motores de Videojuego

A la hora de desarrollar un videojuego debe seleccionarse el motor con el que se va a trabajar. Un motor de videojuegos es una herramienta que provee determinados componentes y rutinas de programación para facilitar el desarrollo, el diseño, y también la ejecución del juego.

Entre las prestaciones que los motores de videojuegos brindan cabe destacar las más importantes, entre ellas el editor de escena, inspector de objetos, administración de recursos, luces, manejo de físicas, materiales, texturas y shaders, editores de terrenos y scripting.

Todos los objetos 3D cuentan con 3 transformaciones posibles: posición, rotación, y escala. En todos los casos se manejan los ejes X, Y y Z.

Si bien los motores gráficos permiten el diseño de escenas robustas, llenas de detalles, partículas y contenidos varios, ninguno presenta la posibilidad de poder editar los objetos desde el punto de vista del diseño, es decir, las caras, lados y vértices de los objetos no pueden ser editados desde el motor, por lo cual, se requiere alguna herramienta adicional de diseño 3D para llevar a cabo esta tarea.

Para la selección de los motores más populares se utilizó Google Trends [1], que es una herramienta que permite obtener estadísticas sobre los datos registrados en el motor de búsqueda de Google. Ésta permite realizar gráficos tanto lineales como de geolocalización sobre términos de búsquedas de los usuarios en un rango configurable de tiempo. Los números reflejan el interés de búsqueda en relación con el valor máximo de un gráfico en una región y un periodo determinados. Un valor de 100 indica la popularidad máxima de un término, mientras que 50 y 0 indican que un término es la mitad de popular en relación con el valor máximo o que no había suficientes datos del término, respectivamente. (ver **Figura 3**).

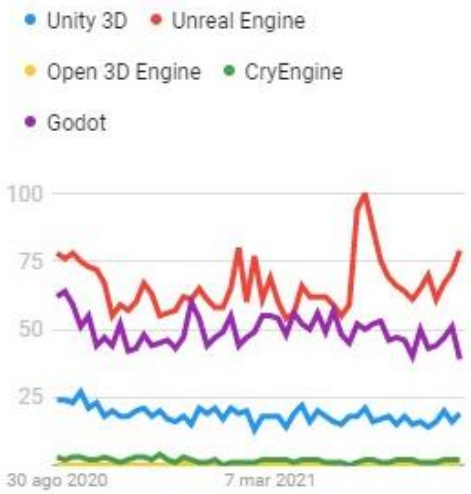

Figura 3. Estadísticas de Consulta de los Motores Gráficos.

A continuación, se detallan los motores más populares del mercado.

2.1. Unity 3D

Unity 3D [17] es uno de los motores de videojuegos más populares del mercado, permite importar una gran cantidad de recursos gráficos (.bmp, .jpg, .png, .psd), modelos 3D (.blend, .fbx, .max, .3DS y obj). Cabe destacar que los archivos correspondientes a Blender [18] y 3DS Max [19] son interpretados de forma nativa por Unity, permitiendo gran facilidad para implementar modificaciones, pero con la contrapartida es que estos archivos son mucho más pesados.

Unity posee un hub donde brinda la opción de instalar diferentes versiones de la herramienta, como también incorporar diferentes módulos, acceder y descargar proyectos de aprendizaje, y centralizar todos los proyectos sin importar la ruta en donde se encuentran, además permite la conversión de cada trabajo a otras versiones, tanto superiores (upgrade) como inferiores (downgrade).

Unity permite al usuario organizar y acomodar los elementos de pantalla de forma irrestricta, facilitando la tarea de diseño.

La pantalla está compuesta por los elementos que se detallan a continuación:

1. Editor de escena: es el lugar donde se diseñan las pantallas, permitiendo agregar, posicionar, rotar y escalar todos los objetos y elementos, como también establecer la iluminación, partículas y efectos.
2. La pestaña de juego (Game): es el sector en el cual se ejecuta el juego, permitiendo realizar las pruebas necesarias y ajustar el diseño.
3. La pestaña de Proyecto: En este sector se agrupan y organizan todos los recursos que pueden agregarse al juego, desde texturas, sprites, modelos, sonidos, etc.
4. La pestaña de jerarquía: Contiene todos los recursos agregados en la escena. Permite

seleccionarlos desde este lugar como también desde el diseñador. Además, se utiliza para emparentar objetos.
5. Inspector: es una pestaña donde se exponen todos los atributos públicos del objeto seleccionado. Desde este lugar es donde se puede establecer valores a cada característica.
6. Console: esta pestaña se utiliza para fines de testeo y detección de errores.

Unity 3D ofrece nativamente objetos a partir de los cuales se puede comenzar a diseñar una escena. Estos objetos se conocen como "GameObjects", y pueden diferenciarse en cubos, esferas, cápsulas, planos y Quads (elementos que siempre pueden dividirse en triángulos).

Además de los componentes primitivos Unity ofrece un diseñador de terrenos muy interesante, que brinda la posibilidad de crear escenarios de una forma muy simple, permitiendo pintar con diferentes texturas los terrenos, establecer los mosaicos de repetición, como también elevar y reducir la superficie del terreno trabajado.

Además, cuenta con funcionalidades para agregar de forma manual y aleatoria diferentes elementos de vegetación, tanto propios como de terceros, como ser pasto, árboles y diferentes tipos de flora. Cada árbol que se planta con la herramienta se agrega de forma automática su respectivo Collider que son elementos de diferentes formas (cubos, prismas, cápsulas, etc.) cuyo volumen se utiliza para sensar la colisión, facilitando la tarea de diseño del escenario.

El diseñador 3D de Unity permite trabajar en dos modos, perspectiva e isométrico, la diferencia entre ambos es que éste último no tenga en cuenta las profundidades ni las proporciones.

Las cámaras dentro de Unity son las responsables de capturar la parte de la escena que se va a renderizar, además, permite indicar en qué lugar y proporción de la pantalla la representación ocupará, posibilitando tener más de una cámara dentro de la misma escena.

Las cámaras pueden configurarse para renderizar la escena en modo "perspectiva" o en modo "ortográfico", la diferencia entre ambos es que éste última ignora las profundidades convirtiéndose en una cámara ideal para el desarrollo de juegos 2D, por consiguiente, la cámara en modo perspectiva es la adecuada para juegos 3D y 2.5D.

Otro elemento importante dentro de los juegos 3D es el manejo de las luces, para ello, cuenta con un componente llamado "Light" en cuatro modalidades:

1. Direccional: Es la luz del ambiente
2. Point: realiza un punto de luz en todas las direcciones
3. Spot: se utiliza para emular la luz de las linternas, es un punto cónico de luz que se agranda con la distancia.
4. Área: establece el reflejo de una luz dentro del volumen de un cubo. Se utiliza para simular la iluminación del sol que ingresa por las ventanas.

Las luces también pueden ser configuradas como elementos que se calculan en tiempo real (consumiendo mayor procesador), o bien ya estar procesada de forma estática (baked), y también en un modo mixto.

Para trabajar con las mecánicas se debe hacer mención sobre el tratamiento de scripts, donde Unity utiliza el paradigma de programación orientada a agentes. Si bien no posee un dentro del aplicativo un IDE de desarrollo de scripts, lo que demanda el uso de aplicaciones externas para compensar esta carencia. Como principal ventaja permite configurar cualquier IDE de desarrollo, por defecto al instalar Unity instala Visual Studio Community [20], pero se puede prefijar cualquier otro IDE como ser Visual Studio Code [20] que es otra una versión liviana de Visual Studio pero que no tiene componentes de diseño visual, otra opción es utilizar MonoDevelop [22], es otro IDE que permite la escritura sencilla de código porque tampoco dispone de una interfaz visual, obteniendo un editor liviano.

Independientemente del IDE de programación, se utiliza únicamente el editor de texto, es decir, no se compila en script en el IDE, sino, al retornar el foco a Unity, automáticamente se realiza la compilación, en el caso de existir algún error en el código se verá reflejado en la pestaña de consola.

Para manejar las físicas Unity 3D provee un control conocido como "Rigidbody" que permite establecer valores de gravedad, restricciones, masa permitiendo simplificar la detección de colisiones con los diferentes objetos, como también las consecuencias del choque de los mismos. Para ello, todos los objetos deben tener sus respectivos "Collider".

Para lograr efectos realistas Unity provee la posibilidad de trabajar con diversos tipos de materiales renderizables, que logran mediante configuraciones efectos 3D con texturas, basados en superposición de imágenes y Shaders, en tendiendo por Shader como un script que indica como debe comportarse la luz al impactar sobre una determinada superficie.

Unity 3D permite compilar sus juegos para Windows 32 y 64 bits, Windows Universal, Linux,MacOs, Android, iOS, WebGL, PS4, PS5, Xbox One, TVOs, Nintendo Switch, y Stabia.

2.2. Unreal Engine

Unreal Engine [23] es uno de los más populares motores de videojuegos que fue desarrollado por la empresa Epic Games [24], su origen fue en el año 1998 presentado en un juego del tipo shooter (juego de disparos en primera persona). A partir de mayo del 2015 la empresa permitió el uso gratuito del motor.

Actualmente, la versión actual de Unreal Engine es la número 5. Entre sus características más destacadas podemos mencionar un HUB que permite la descarga de proyectos de prueba, la instalación de recursos y librerías, la creación de nuevo proyectos basados en plantillas, Como también la instalación de nuevas versiones y nuevos compilados del del motor. Además, brinda la posibilidad de

poder desarrollar videojuegos con diferentes versiones siendo estas administradas desde el HUB.

Unreal Engine cuenta con un IDE de desarrollo muy robusto, al igual que el resto de los motores existentes en el mercado posee pestañas claramente identificables que integran todos los elementos gráficos y propiedades para él diseño, desarrollo, pruebas y compilación de escenas.

La interfaz gráfica se posee las siguientes características:

1. Pantalla de diseño.
2. Cuadro de herramientas.
3. Barra de herramientas.
4. Explorador de objetos (World Outliner).
5. Pestaña de propiedades.
6. Pestaña de contenidos.

El diseñador de Unreal Engine ofrece una numerosa cantidad de objetos predefinidos para comenzar a trabajar, separado por categorías que facilitan la usabilidad del IDE.

Al igual que el común denominador de los motores gráficos Unreal Engine presenta su propio diseñador de terrenos, posee la capacidad de elevar, o descender superficies, incorporar vegetación y follaje, utilizando herramientas prediseñadas como también personalizadas.

Unreal Engine también presenta un sistema de iluminación con cinco tipos de luces diferentes:

1. Directional Light: Proyecta una luz de una fuente muy lejana como ser el sol, y las sombras que se emiten son paralelas a dicha fuente.
2. Sky Light: Permite la iluminación de una escena a partir de las imágenes de fondo.
3. Spot Light: Emite luz en forma de cono, permitiendo al usuario regular los ángulos de brillo y alcance.
4. Point Light: Son emisiones de luz desde un punto específico que ilumina un radio determinado.
5. Rect Lights: emite luz en la escena desde un plano rectangular con un ancho y una altura definidos.

En cuanto a las físicas Unreal Engine maneja los objetos Collision, que disparan eventos cuando se detecta colisión de algún tipo, para ello, se debe configurar el objeto como Physicsbody (cuerpo físico). A diferencia de otros motores, el elemento RigidBody se utiliza para establecerlo en personajes animados para poder simular las leyes de la física.

Otro aspecto para destacar de este motor son las plataformas de destino que corren los juegos: Windows 32 y 64 Bits, MacOS, Linux PCs, PS4, PS5, Xbox One, Xbox Series X, Nintendo Switch; iOS y dispositivos de Android.

2.3. CryEngine

CryEngine [25] es un motor de videojuegos que fue lanzado al mercado en el año 2002 por la empresa alemana Crytek [26].

A diferencia de otros motores Cryengine permite importar un número reducido de recursos externos, entre ellos modelos 3D (.fbx, .dxf, .dae, .obj, y .3ds), para archivos gráficos (.bmp, jpg, png, .ico). Al inicializar la herramienta se puede apreciar que posee un hub en donde se puede instalar diferentes versiones del motor, como también centralizar los proyectos y recursos, además, permite recibir noticias sobre diferentes productos y acceder al Marketplace de la empresa. CryEngine permite personalizar el espacio de trabajo, dejando libre albedrío para acomodar las pestañas al lugar de la pantalla que se desee, como también las barras de controles.

La interfaz de usuario se la conoce como Sandbox y provee las siguientes funcionalidades básicas:

1. Pestaña de creación de objetos.
2. Explorador del Nivel.
3. Viewport de diseño y ejecución.
4. Explorador de recursos.
5. Pestaña de propiedades.
6. Editor de Terreno.

A la hora de importar modelos 3D, CryEngine realiza una adaptación de este recurso dentro al entorno para poder utilizarse, requiriendo un tiempo considerablemente largo para realizar la dicha adaptación. Como contrapartida a mayor cantidad de recursos que se importan, mayor la pérdida de tiempo.

CryEngine provee un robusto diseñador de terreno, con herramientas para elevar, descender y suavizar terrenos, como también de creación de huecos y rellenos.

Por defecto, establece una base con superficie de agua que abarca toda la escena. A pesar de la interfaz robusta y de la posibilidad de utilizar el mouse para diseñar los escenarios, la usabilidad se ve opacada dado que la manipulación de elementos, como también los movimientos dentro de la escena son lentos y poco fluidos, como también gran parte de las acciones que se solicitan demoran segundos en realizarse, dado que consume muchos recursos de memoria, disco y video.

El sistema de cámaras proporcionado permite únicamente cámaras con vista en perspectiva, pero como punto a favor cuenta con una numerosa cantidad de configuraciones que se le pueden aplicar.

Para manejar las luces provee 2 componentes:

1. Light
2. Projector Light

El primer elemento “Light” consiste en incorporar un punto de luz que iluminará un radio específico en todas las direcciones.

Projector Light es un componente que se encarga de generar una proyección de la luz como si de una linterna se tratara, permitiendo configurar el ángulo, la intensidad y el tipo de proyección.

A la hora de trabajar con mecánicas mediante scripts CryEngine provee dos herramientas visuales, la primera es una herramienta de flujo gráfico (Flow Graph Tool) y la segunda es un elemento experimental conocido como

"Schematyc" que permite realizar secuencias de comandos de entidades.

Flow Graph Tool consiste en establecer en una entidad del nivel específica el agregado de diferentes nodos que poseen un comportamiento implícito, de mismo modo, se conecta a las salidas con las entradas de forma visual.

Para poder programar en CryEngine se debe ir al explorador de carpetas y elegir la opción de generar Solución, de allí se crea una solución de Visual Studio para poder realizar la implementación de programación que se considere necesaria.

CryEngine provee un mecanismo de simulación de físicas excelente, permite determinar a una entidad como se debe comportar ante el movimiento y una eventual colisión. Los componentes utilizados para aplicar físicas son "Rigidbody" y "LocalGrid". Éste último establece un subespacio de simulación local en el mundo, en el cual se establece un umbral de aceleración, y de reglas físicas. En cambio, el componente "Rigidbody" permite configurar entre una numerosa cantidad de parámetros la masa, resistencia al aire, y otras variables más.

En cuanto a las plataformas de destino, CryEngine permite desarrollar juegos para Windows, Linux, PlayStation 4, Xbox One, Oculus Rift, OSVR, PSVR, HTC Vive, Android e IOs.

2.4. Godot Engine

Godot Engine [27] Es un motor gráfico muy liviano, que no requiere instalación, A diferencia de los otros motores no dispone de un HUB para poder administrar las versiones del motor. En cambio, provee una pestaña que permite descargar plantillas de diferentes estilos de juego, y recursos.

El editor de Godot cuenta con:

1. Viewport para el diseño y ejecución
2. Panel de escena
3. Pestaña de FileSystem
4. Pestaña de Inspector
5. Barra de herramientas, dependiendo del tipo de juego 2D o 3D las opciones cambian
6. Sección de propiedades

Por defecto Godot no posee una herramienta que le permita editar terrenos, no obstante, requiere de la instalación adicional de un plugin para poder incorporar esta característica. Dicho plugin permite adquirir las funcionalidades de modelar terrenos, elevando, suavizando, pintando las superficies como también agregar vegetación.

La interfaz de usuario permite la manipulación de los recursos mediante el uso del Mouse, facilitando de esa manera el diseño de las escenas, proporcionando buena usabilidad.

Dentro de Godot se utiliza el concepto de "Viewport" para indicar como se renderiza una escena, siendo el equivalente a las "Cámaras" de los otros motores. También permite renderizar en elementos específicos conocidos como "texturas dinámicas", que proyectan una vista sobre una superficie de un objeto, por ejemplo, un espejo.

Para la iluminación Godot maneja 4 tipos de nodos de luz:

1. Directional: Es un tipo de luz que proviene de un origen lejano, por ejemplo, el sol.
2. Omni: Este tipo de luz se emite hacia todas las direcciones de manera esférica.
3. Spot: Es un tipo de luz focal que se emite en forma de cono, basada en un ángulo de proyección.
4. Environmet: corresponde a la luz ambiental producida por el skybox seleccionado.

Para el desarrollo de las mecánicas Godot ofrece varias alternativas para la programación, La primera es GDScript que es un lenguaje de programación de alto nivel con una sintaxis parecida a Python. Visual Scripting es otra alternativa que no demanda programación por código, sino que establece un mecanismo de lógica de flujos ideal para aquellas personas que no son programadoras. Por medio del uso de Visual Studio como herramienta externa Godot permite programar en C#. Como un detalle adicional Godot permite el cruce de lenguajes, permitiendo a todos los scripts colaborar entre si independientemente del lenguaje en el que están hechos.

Para la aplicación de las físicas Godot provee 4 objetos del tipo Colisión:

1. Area2D: son nodos de detección e influencia, permite detectar superposición de objetos, entradas y salidas entre ellos, y reemplaza a las leyes de la física.
2. StaticBody2D: Este elemento no puede ser movido por ningún elemento del motor de física. Se utiliza para elementos que no requieren comportamiento dinámico.
3. RigidBody: Representa un cuerpo físico y emula su comportamiento por medio del motor de físicas, por ejemplo, se aplica la ley de gravedad.
4. KinematBody: Es un cuerpo físico pero que no le afecta la ley de gravedad.

Cada uno de ellos detectan las colisiones mediante los objetos del tipo Collision correspondiente.

Las plataformas de destino para los videojuegos hechos en Godot son: Windows, Linux, Mac Os, Android y Web Assembly.

2.5. Open 3D Engine

Amazon Lumberyard [28] es un motor de videojuegos que puso su código fuente disponible y pasó a ser O3DE [29] desarrollado por la empresa Amazon Inc. [30].

A diferencia de otros motores no cuenta con un hub centralizado para la instalación del motor, de hecho, la instalación del motor lleva varios pasos de forma manual, entre ellos, la descarga de las fuentes de un repositorio GIT, la instalación de herramientas adicionales, Visual Studio 2019 [19], y CMake [31], y por último la descarga de paquetes y compilación del motor gráfico. Esta numerosa

cantidad de pasos a realizar demandan controlar los privilegios de acceso.

El diseñador cuenta con las siguientes partes:

1. Asset Browser: contiene todos los recursos para el diseño de escenas.
2. Entity Outliner: en este apartado están todos elementos que se encuentran incorporados a la escena.
3. Entity Inspector: contiene las propiedades y características de cada objeto.
4. ToolBox: posee una caja de herramientas para la manipulación de los elementos muy reducida.
5. Diseñador: es el espacio donde se agregan los objetos y construyen las escenas

Para diseñar los terrenos O3DE utiliza una herramienta internar que permite modelar los terrenos, pero a diferencia de otros motores, el diseñador no es sobre el terreno 3D, sino que, por el uso de imágenes, el terreno se divide en mosaicos de igual tamaño a los que se les puede configurar diferentes elementos. Se establecen texturas 2D para indicarla elevación y los detalles, y para implementar vegetación se hace el uso de una GEM (addon de propio motor), que permite distribuir vegetación a lo largo del terreno.

Al igual que el resto de los motores O3DE posee Cámaras que periten renderizar las escenas, o parte de ellas. Además, provee un componente que le permite agregar movimientos especiales a la cámara.

Los componentes de luces que ofrece el motor son:

1. Area Light: Es un tipo de luz que ilumina el interior del volumen de un cubo.
2. Point Light: Este tipo de luz se emite hacia todas las direcciones de manera esférica.
3. Proyector Light: Es un componente que permite a una entidad proyectar luz.
4. Environmet Probe: corresponde a la luz ambiental producida por el skybox (textura panorámica que se renderiza detrás de todos los objetos de la escena representando el fondo que rodea al juego) seleccionado.

Dentro de O3DE se utiliza como lenguaje de programación de cabecera LUA [31]. También ofrece una herramienta de programación visual por medio de nodos llamada Script Canvas.

Con respecto a las físicas O3De provee tres componentes:

1. PhysX Terrain: Este componente se aplica en los terrenos, permitiéndole responder a estímulos detectando las colisiones sobre las superficies irregulares.
2. PhysX Collider: Este componente se le aplica a los objetos brindándole la posibilidad de detectar y calcular las colisiones entre entidades. Las entidades simples pueden tener un solo Collider, pero, otras entidades pueden tener múltiples colliders, por ejemplo, un auto.
3. PhysX Rigid Body: convierte a la entidad como un objeto sólido y rígido, que comienza a afectarle las leyes de la gravedad, y puede colisionar con otros objetos.

Para su producción y distribución, Open 3D Engine compila sus juegos para las plataformas Windows, MacOs, Linux, Andriod e IOs.

2.6. Comparativa entre motores

Como se ha expuesto en la sección anterior todos los motores de desarrollo de videojuegos cuentan con prestaciones similares, desde el diseñador gráfico, inspector de elementos, recursos de proyecto, editores de terrenos, y manejo de luces, Shaders, y scripting, cada uno con su respectivo formato de diseño, pero, en resumidas cuentas, todos los motores tienen lo mismo. A la hora de seleccionar un motor gráfico se debe tener en consideración otros criterios que apuntan a la facilidad de uso, documentación y respaldo que el motor presenta. A continuación, se exponen algunos de estos criterios (ver **Tabla 2**). Criterios: CR1: ¿Posee Hub centralizado para versiones del motor, proyectos y recursos?, CR2: ¿Posee una Interfaz amigable?, CR3: ¿Permite Debug en Tiempo de Ejecución?, CR4: Documentación disponible, valores posibles M: Mucha, P: poca, CR5: Costo de licencia, valores posibles G: Gratuita, C: Condicional a la cantidad de ingresos por ventas. CR6: ¿Es software de código abierto?, CR7: Cantidad de plataformas de destino.

Tabla 2. Tabla Comparativa de Motores Gráficos

Motor	CR 1	CR 2	CR 3	CR 4	CR 5	CR 6	CR 7
Unity 3D	Si	Si	Si	M	C	No	13
Unreal Engine	Si	Si	No	M	G	No	10
Cry Engine	Si	No	No	M	C	No	9
Godot	No	Si	No	M	G	Si	5
Open 3D Engine	No	No	No	P	G	Si	5

Además de los criterios ya expuestos, en la **Tabla 3** se comparan los lenguajes de programación que soportan los diferentes motores junto con las plataformas de destino para las que pueden ejecutar los juegos desarrollados.

Tabla 3. Tabla Comparativa de Motores Gráficos según plataformas y lenguajes soportados.

Motor	Lenguajes soportados	Plataformas destino
Unity 3D	C#, Javacript, Boo Script	Windows 32 y 64 bits, Windows Universal, Linux, MacOs, Android, iOS, WebGL, PS4, PS5, Xbox

		One, TVOs, Nintendo Switch, y Stabia.
Unreal Engine	C++	Windows 32 y 64 Bits, MacOS, Linux PCs, PS4, PS5, Xbox One, Xbox Series X, Nintendo Switch; iOS y Android.
Cry Engine	C++, C#	Windows 32 y 64 bits, Linux, PlayStation 4, Xbox One, Oculus Rift, OSVR, PSVR, HTC Vive, Android e IOs
Godot	C#	Windows, Linux, Mac Os, Android y Web Assembly.
Open 3D Engine	LUA	Windows 32 y 64 bits, MacOs, Linux, Andriod e IOs

Por último, cabe destacar los datos reflejados por Google Trends, sobre la intención de consulta en el mundo durante los últimos 12 meses, en los que se incluye la búsqueda de información por navegadores, video streaming, etc. Unreal Engine y Godot son los motores de videojuegos más consultados, por debajo Unity 3D, y con un muy bajo nivel de consulta CryEngine y Open 3D Engine.

3. Relevamiento de Diseñadores 3D para Videojuegos

Para desarrollar videojuegos con utilizar un motor gráfico únicamente no es suficiente, se requieren herramientas adicionales, programas de tratamiento de imágenes, musicalizadores, diseñadores 3D, etc. para poder desarrollar los recursos necesarios que serán importados y gestionados por el motor de videojuegos. En este trabajo se acota el análisis a los programas de modelado 3D.

Como se mencionó anteriormente, los motores requieren de diseñadores gráficos para poder diseñar objetos a medida, que cumplan con los requerimientos que el recurso amerite.

Existen varios diseñadores 3D en mercado, algunos son utilizados para tareas de arquitectura, otros tienen un enfoque para impresoras 3D, en este trabajo se relevaron los 5 más importantes que se utilizan en videojuegos y se detallan sus características a continuación.

3.1. 3D Max Studio

3D Max Studio o también conocido como 3DS Max [18], es una herramienta de modelado 3D, renderización y animación muy popular, desarrollada por la empresa Autodesk Inc.

Provee una gran cantidad de funcionalidades con respecto al modelado, múltiples cámaras que permiten ver desde diferentes ángulos la misma escena. Para facilitar el proceso de modelado brinda la posibilidad de importar blueprints, siendo estos, imágenes en dos dimensiones a escala con varios perfiles del modelo a realizar, que, al ser importados dentro del diseñador, ayudan a mantener la escala y proporciones del modelo desde diferentes vistas mediante el “calcado” digital del dibujo.

Dispone de diferentes objetos primitivos, como también luces, y permite la manipulación de materiales mediante el uso de texturas para lograr efectos realistas.

Los objetos pueden ser manipulados desde sus 3 trasformaciones: posición, rotación y escala, tanto el objeto global, como cada una de sus partes, sean caras, lados o vértices.

Cuenta con una numerosa cantidad de plugins, entre los cuales permite la animación de un modelo digital conocido como “rigging” mediante el uso de un conjunto de huesos conocidos como “Armature”, creación de escenarios con vegetación, etc. Además, permite realizar animaciones de los objetos, ideal para ser utilizados para los diferentes motores gráficos.

La interfaz gráfica es amigable, robusta y cuenta con numerosa cantidad de funciones para cada aspecto del diseño.

La licencia de 3D Max Studio es paga y la aplicación es un enlatado.

3.2. Autodesk Maya

Autodesk Maya [33] es uno de los programas de diseño 3D más potentes del mercado. Es una aplicaciones más robustas y completas desarrollada por la firma Autodesk Inc.

La herramienta provee varios objetos primitivos, como permite la manipulación de materiales mediante el uso de texturas para lograr efectos realistas, y dispone de unos elementos de luz que resaltan la calidad del renderizado. También dispone de varios plugins que amplían las funcionalidades del motor.

Al igual que 3D Max Studio, Maya permite la animación y control de personajes (rigging), utilizando el componente “Armature” (esqueleto utilizado para los movimientos de los modelos) y el uso de blueprints..

En cuanto a las vistas, la pantalla puede dividirse para enfocar una escena desde diferentes ángulos en simultáneo

Además, todos los objetos soportan las 3 transformaciones (rotación, posición y escala), desde el objeto general, pasando por las caras del elemento, lados y vértices.

La herramienta no es de código abierto, y su licencia de uso es de pago.

3.3. Blender Studio

Es uno de los diseñadores 3D más populares, es una herramienta open source escrita en python, que permite mediante una interfaz robusta modelar objetos 3D.

Ofrece una cantidad objetos primitivos, como también luces y efectos de renderizado. Los de materiales pueden

manipulares mediante el uso de texturas para lograr efectos realistas.

Posee una comunidad muy grande de desarrolladores que lo utilizan, y al ser de código abierto la cantidad de funcionalidades adicionales es muy grande. Posee muchos agregados, tanto para "rigging", creación de terrenos, importación de ciudades a partir de un mapa de Open Street Maps, etc.

Los personajes pueden ser fácilmente diseñados dado que permite la importación de blueprints, también posee herramientas para el rigging de "Armatures" y la animación de este.

Todos los objetos pueden ser manipulados en todas sus transformaciones, desde el objeto general, como también, las caras, lados y vértices, permitiendo tener elementos completamente personalizados.

3.4. 3D Builder

3D Builder [34] es una herramienta desarrollada por Microsoft que permite el modelado de objetos 3D a partir de combinar objetos primitivos provistos por la aplicación. Se orienta especialmente para el diseño de partes para la impresión en impresoras 3D dado que admite los formatos de archivo de impresión 3D más importantes: STL, OBJ, 3MF, etc.

En cuanto al modelado la conformación de nuevos elementos se establece mediante la unión y sustracción de intersecciones de los objetos, aunque no permite modificar los lados de forma individual como tampoco los vértices, únicamente las caras de la forma pueden ser trasladas. La rotación y escala quedan suscriptas al objeto entero.

A diferencia de otros diseñadores, 3D Builder no cuenta con funciones de animación, como tampoco la posibilidad de realizar rigging de personajes, tampoco provee materiales para la aplicación de texturas.

Es una aplicación enlatada que se encuentra disponible para su descarga en la tienda de Windows, y su licenciamiento es gratuito,

3.5. LeoCAD

LeoCAD [35] es una herramienta que fue desarrollada basándose en el juego de bloques Lego, donde por medio de una gran cantidad de piezas de encastre, permite realizar construcciones encastrando las mismas. Este diseñador utiliza el mismo criterio del juego físico, mediante bloques virtuales que se conectan entre sí, permitiendo la incursión de usuarios no familiarizados con herramientas de diseño 3D. Los bloques no son personalizables, tienen una forma fija y tamaño que no puede ser alterado.

La plataforma proporciona más de 10.000 partes diferentes, pero también cuenta con una biblioteca de partes descargables desde LDraw [36].

La herramienta cuenta con la posibilidad de animar los bloques realizando traslados, rotación y escala, pero no dispone de ningún tipo de rigging para animar personajes.

En cuanto al licenciamiento es un software de modelado 3D gratuito. LeoCAD es de código abierto y está disponible para Windows, Linux, macOS.

3.6. Análisis Comparativo de Diseñadores 3D

A continuación, se reflejan diversas comparativas entre los diseñadores 3D expuestos. En primer lugar, se detalla mediante el uso de Google Trends el interés de búsqueda y consulta a lo largo del tiempo.

Las tendencias recogidas de la herramienta Google Trends [16] (**Ver Figura 4.**) demuestran que las personas a lo largo de los últimos 12 meses en el mundo realizaron mayor cantidad de búsquedas de información (en las que se incluyen documentos, videos de streaming, etc.) sobre el diseñador Autodesk Maya, Blender Studio, 3D Max Studio y 3D Builder están ampliamente por debajo, y LeoCad es el diseñador que refleja menor interés de consulta.

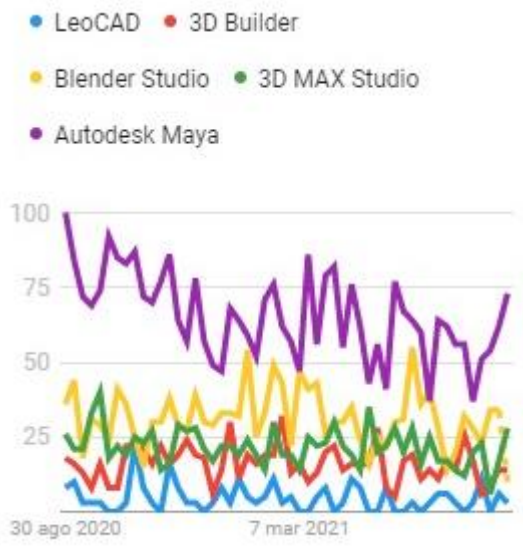

Figura 4. Estadísticas de Consulta de los Diseñadores 3D.

En segundo lugar, se detalla en la Tabla 4 los criterios comparativos entre los distintos diseñadores. Criterios: CR1: Permite el modelado personalizado de elementos CR2: ¿Posee una Interfaz amigable?, CR3: ¿Permite desarrollar animaciones?, CR4: Documentación disponible, valores posibles M: Mucha, P: Poca, CR5: Costo de licencia, valores posibles G: Gratuita, P: de Pago. CR6: ¿Es software de código abierto?, CR7: ¿Permite Múltiples vistas?

Tabla 4. Tabla Comparativa de Diseñadores 3D.

Diseñador	CR 1	CR 2	CR 3	CR 4	CR 5	CR 6	CR 7
3D Max Studio	Si	Si	Si	M	P	No	Si
Autodesk Maya	Si	Si	Si	M	P	No	Si
Blender Studio	Si	Si	Si	M	G	Si	Si
3D Builder	No	Si	No	M	G	No	No

LeoCad	No	Si	Si	P	G	Si	No

4. Conclusiones

En el presente trabajo se realizó el relevamiento y posterior análisis de los diferentes motores gráficos de videojuegos, como también de los diseñadores 3D. Se establecieron los elementos comunes de cada una de las herramientas como también se expusieron los aspectos en que se diferencian cada uno de ellos. Estos criterios evaluados presentan un aporte importante a la hora de decidir que herramientas utilizar cuando se pretende desarrollar un videojuego, donde el presupuesto puede ser un factor determinante para la realización del mismo, por lo que se puede afirmar que para lograr producir un videojuego de calidad sin presupuesto para los motores y diseñadores, Unreal Engine como primera opción, Godot en Segundo lugar y O3De al final junto con Blender Studio serían las opciones más adecuadas para las plataformas de destino comunes (Windows, Linux, MacOs, Android e Ios).

En un segundo escenario donde se cuente con presupuesto para invertir en herramientas Unreal Engine y Unity serían las opciones más adecuadas siendo que éste último, su licencia comienza abonarse a partir de una base de ganancias, la siguiente opción sería CryEngine, con el mismo modelo de licenciamiento que Unity pero su interfaz es menos amigable y cuenta con mucha menos documentación, la elección de estos motores permiten invertir en diseñadores como 3D Max Studio y Autodesk Maya, sin descartar a Blender.

El último escenario de presupuesto ideal permite seleccionar en primer lugar a Autodesk Maya como diseñador de recursos, por su usabilidad, cantidad de documentación y servicios posventa de soporte y consulta, y como motor gráfico Unreal Engine en primer orden y Unity 3D en segundo lugar. En caso de que el requerimiento sea abarcar la mayor cantidad de plataformas Unity sería la opción más adecuada, dado que es el motor que más plataformas soporta.

Referencias

[1] Lacasa, P. (2011). Los videojuegos: aprender en mundos reales y virtuales. Los videojuegos, 1-327.

[2] Domínguez, F. I. R., & Antequera, J. G. (2012). ¿Qué aprendo con videojuegos? Una perspectiva de meta-aprendizaje del videojugador. Revista de Educación a Distancia (RED), (33).

[3] Eguía, J. L., Contreras, R. S., & Solano Albajés, L. (2012). Videojuegos: conceptos, historia y su potencial como herramientas para la educación.

[4] Marqués, P. (2001). Los videojuegos. Aula de Innovación Educativa.

[5] Wolf, M. J., & Perron, B. (2005). Introducción a la teoría del videojuego. Formats: revista de comunicación audiovisual.

[6] Conde, M. E., & Artola, E. L. (2017). Experiencias de programación de videojuegos en diferentes escenarios socioculturales.

[7] Bethke, E. (2003). Game development and production. Wordware Publishing, Inc.

[8] Alberto Arjona Molina (2019). GDD - El documento de diseño, Game Design Document. Disponible en: https://aprendiendoadesarrollarvideojuegos.blogspot.com/2019/07/gdd-el-documento-de-diseno.html

[9] Pereira, A. M. M. (2014). El proceso productivo del videojuego: fases de producción/The production process of the game: production phases. Historia y comunicación social, 19, 791-805.

[10] Latorre, Ó. P. (2011). Géneros de juegos y videojuegos: una aproximación desde diversas perspectivas teóricas. Comunicación: revista de recerca i d'anàlisi, 127-146.

[11] Sarango, Y. E. L. (2013). Difusión de los géneros, características, plataformas, herramientas de desarrollo que se usan en la industria de los videojuegos. Revista Tecnologica - ESPOL, 26(2).

[12] Wolf, M. J. (2001). The medium of the video game. University of Texas Press.

[13] Wolf, M. J., & Perron, B. (2005). Introducción a la teoría del videojuego. Formats: revista de comunicación audiovisual.

[14] Hanna, P. (2012). Video game technologies. Java Games Programming.

[15] Santos, A. F., Alloza, S., & Escribano, F. (2018). Manual para educadores: Relación entre géneros de videojuegos y soft skills. Recuperado de https://gecon. es/wp-content/uploads/2018/04/gecon. es-Genero_videojuegos_soft_skills. pdf.

[16] Google trends. (2021). Recuperado el 1 de 9 de 2021, de https://trends.google.es/trends/?geo=AR

[17] Unity. (2021). Recuperado el 1 de 9 de 2021, de https://unity.com/

[18] Blender. (1 de 9 de 2021). Obtenido de https://www.blender.org/

[19] 3DS Max. (s.f.). Obtenido de https://latinoamerica.autodesk.com/products/3ds-max/overview

[20] Visual Studio Comunity. (2021). Recuperado el 1 de 9 de 2021, de https://visualstudio.microsoft.com/es/vs/community/

[21] Visual Studio Code. (2021). Recuperado el 1 de 9 de 2021, de https://code.visualstudio.com/

[22] MonoDevelop. (2021). Recuperado el 1 de 9 de 2021, de https://www.monodevelop.com/

[23] Unreal Engine. (2021) Recuperado el 26 de 9 de 2021. Obtenido de The most powerful real-time 3D creation platform - Unreal Engine: https://www.unrealengine.com/

[24] Epic Games. (2021). Recuperado el 26 de 9 de 2021. Obtenido de Epic Games: https://www.epicgames.com/site/es-ES/home

[25] Cryengine. (2021). Recuperado el 1 de 9 de 2021, de Cryengine -The complete solution for next generation games: https://www.cryengine.com/

[26] Crytek. (2021). Recuperado el 1 de 9 de 2021, de Crytek videogame developers. Makers of CryEngine: https://www.crytek.com/

[27] Godot. (2021). Recuperado el 1 de 9 de 2021, de Godot Engine - Free and opem source 2D and 3D game engine: https://godotengine.org/

[28] AWS Amazon Lumberyard. (2021). Recuperado el 1 de 9 de 2021, de Motor de videojuegos completamente personalizable: https://aws.amazon.com/es/lumberyard/

[29] Open 3D Engine. (2021). Recuperado el 1 de 9 de 2021, de O3DE: https://o3de.org/

[30] Amazon. (2021). Recuperado el 1 de 9 de 2021, de Amazon - Gasta menos. Sonríe más: https://www.amazon.com/

[31] CMAKE. (2021). Recuperado el 1 de 9 de 2021, de CMAKE: http://cmake.org

[32] Lua. (2021). Recuperado el 1 de 9 de 2021, de https://www.lua.org/

[33] Autodesk Maya Studio (2021). Recuperado el 1 de 9 de 2021, de https://www.autodesk.com/products/maya/

[34] 3D Builder. (2021). Recuperado el 1 de 9 de 2021, de https://www.microsoft.com/es-ar/p/3d-builder/9wzdncrfj3t6?activetab=pivot:overviewtab

[35] LeoCad. (2021). Recuperado el 1 de 9 de 2021, de LegoCad - Virtual LEGO CAD Software: https://www.leocad.org/

[36] LDraw.org (2021). Recuperado el 1 de 9 de 2021 – de LDraw.org Centralized LDraw Resources https://www.ldraw.org/

9° CoNaIISI
2021

Congreso Nacional de
Ingeniería Informática y
Sistemas de Información