TecsMedia

Análisis: Motores gráficos y su aplicación en la industria

GOBIERNO DE ESPAÑA
MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD

GOBIERNO DE ARAGON
Departamento de Innovación, Investigación y Universidad

TecsMedia

Documento generado por técnicos de la División de Tecnologías Multimedia del Instituto Tecnológico de Aragón (ITAINNOVA) enmarcado en la **ORDEN de 29 de diciembre de 2014, del Consejero de Industria e Innovación, por la que se encomienda al Instituto Tecnológico de Aragón la realización de actuaciones para potenciar e impulsar el sector de tecnologías audiovisuales en Aragón.**

ITAINNOVA
INSTITUTO TECNOLÓGICO DE ARAGÓN

Material desarrollado por el Instituto Tecnológico de Aragón

Referencia / Cita:
"González Muñoz C., Gracia Bandrés, M.A., Sanagustín Grasa, L., Romero San Martín, D. – (2015) TecsMedia: Análisis Motores gráficos y su aplicación en la industria"

www.aragon.es
www.itainnova.es

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD

GOBIERNO DE ARAGON
Departamento de Innovación, Investigación y Universidad

Índice

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD

GOBIERNO DE ARAGON
Departamento de Innovación, Investigación y Universidad

01. Resumen ejecutivo

Los motores gráficos son entornos de trabajo diseñados específicamente para la creación y desarrollo de video juegos. En sus orígenes las arquitecturas software empleadas para el desarrollo de video juegos no establecían una separación clara entre distintos componentes y fue a mediados de los 90 cuando el desarrollo del juego Doom mostro una arquitectura que permitía la reutilización de muchos componentes.

A partir de ese momento se empezó a acuñar el término “Game Engine” para referirse a los componentes comunes sobre los que se asientan nuevos desarrollos. Tal ha sido su evolución y optimización que los motores gráficos están experimentando una evolución desde el campo del entretenimiento puro a nuevos usos orientados a otros entornos o sectores, como pueden ser la formación y entrenamiento a través del desarrollo de simuladores, que es lo que se denomina Serious Games, u otro tipo de aplicaciones de Realidad Virtual enfocadas al sector industrial.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON
Departamento de Innovación, Investigación y Universidad

02. Introducción a los motores gráficos

Se define como motor gráfico al framework de software **diseñado para crear y desarrollar videojuegos**. Los desarrolladores de videojuegos pueden usar los motores para crear videojuegos para consola, dispositivos móviles, ordenadores o dispositivos de Realidad Virtual.

Se puede establecer el origen del término "*Game Engine*" a mediados de los años 90 con la aparición del juego de primera persona Doom (de la Empresa "Id Software") por la arquitectura software del mismo. Con la realización de este juego, se separaron los sistemas de renderizado gráfico, de detección de colisiones, audio, objetos de juego y reglas. Las ventajas de esta separación en componentes se hicieron evidentes en juegos posteriores al poder reutilizar componentes en nuevos desarrollos (Quake III, Unreal, etc.)

Todo motor gráfico ha de ofrecer al programador una funcionalidad básica, proporcionando normalmente un motor de **renderizado** ("render") para gráficos 2D y 3D, un motor que detecte la **colisión física de objetos** y la respuesta a dicha colisión, sonidos y música, animación, inteligencia artificial, comunicación con la red para **juegos multijugador**, posibilidad de ejecución en hilos, gestión de memoria o soporte para localización. Si bien en sus orígenes el desarrollo de videojuegos era muy dependiente de la plataforma, la evolución natural ha sido la de ofrecer un entorno de desarrollo que permitiera portar los desarrollos a distintas plataformas.

Tradicionalmente, estos motores han tenido diferencias entre los distintos géneros de juegos:

- **Juegos en primera persona (FPS)** donde el mayor reto se centra en ofrecer entorno hiper realistas con renderizado 3D eficiente de amplios espacios; control preciso de cámara; Inteligencia Artificial (AI) para dar vida a los jugadores no reales (*Non Player characters* (NPCs) o Entorno Multijugador

- **Juegos de plataforma y de tercera persona.** Los juegos de plataforma (*Platformers*) incluyen títulos clásicos como *Donkey Kong*, *Super Mario Brothers*. Al abordarse juegos 3D como Super Mario 64, Sonic the Hedgehog, etc.; se vió que los requisitos técnicos eran similares a los juegos de tercera persona, por lo que se presentan agrupados. Este grupo de juegos pone el énfasis en los movimientos del personaje, cámaras, etc.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON Departamento de Innovación, Investigación y Universidad

03. Motores de juegos o game engines

En esta sección, se incluyen los distintos tipos de game engines o motores de juegos principales:

03.1. CryEngine

Nació como un demostrador de la empresa *Crytek* de las capacidades de la tarjeta gráfica Nvidia evolucionando hacia un juego completo llamado *Far Cry*. CryEngine3 se ha convertido en una plataforma para desarrollo de juegos con herramientas para la creación de assets y gráficos de gran calidad en tiempo real. Los desarrollos son multiplataforma incluyendo *Xbox One, Xbox 360, PlayStation 4, PlayStation 3, Wii U y PC.*

03.2. Unity

Entorno de desarrollo de juegos multiplataforma, así como su runtime que permite desplegar los desarrollos a un gran número de plataformas incluyendo las móviles como Apple iOS, Google Android, Windows Phone, Blackberry 10, consolas (*Microsoft Xbox 360, Xbox One, Sony Playstation 3 y 4, Nintendo Wii* y *Wii U*) así como entornos de escritorio (*Microsoft Windows, Apple Macintosh* y *Linux*) o incluso un *Webplayer* para el despliegue en los principales Navegadores.

El entorno unificado de desarrollo permite desde la creación, manipulación y visualización de assets y entidades, visualización tanto en el escritorio como en el hardware destino (móvil conectado al USB en modo depuración). Ofrece también herramientas para analizar el comportamiento en las distintas plataformas y permite la programación en lenguajes JavaScript, C# o Boo para facilitar la creación de animaciones.

03.3. Unreal Engine

Nacido en 1998, Epic Games empleó este motor para desarrollar *Unreal*. Desde entonces se ha convertido en un referente en muchos aspectos como en los juegos de primera persona (*FPS*). *Unreal Engine 4* (*UE4*) es la última versión y destaca por herramientas para la creación de *Shaders*, programación de lógicas de juego y gran entorno gráfico de desarrollo.

Su arquitectura abierta permite la modificación del motor para adaptarse a necesidades específicas de desarrolladores para optimizar su funcionamiento en distintas plataformas y destaca en el desarrollo de juegos 3D de primera persona.

Se trata de una plataforma de pago si bien mucha de su documentación está disponible de forma gratuita y por una pequeña cuota da acceso al código fuente del motor y a toda la documentación lo que la presenta como una alternativa viable a pequeños desarrolladores.

03.4. Microsoft XNA

Basado en lenguajes .NET manejados, con el soporte de todas las librerías genéricas de .NET y algunas librerías específicas XNA, es una plataforma enfocada a usuarios con un nivel medio de programación. Se desarrolla sobre Microsoft Visual Studio desde la creación de *Assets* hasta la programación y permite desarrollar aplicaciones para PC y para la consola Xbox 360. Existe una plataforma de comercialización de estas aplicaciones en la que se pueden publicar los desarrollos propios por una pequeña tasa, lo que ha atraído a muchos desarrolladores.

03.5. Sony PhyreEngine

En el año 2008, Sony presentó esta plataforma para hacer más accesible el desarrollo de juegos para sus consolas. *PhyreEngine* se ha convertido en la herramienta de referencia para la creación de juegos para la PlayStation 4, PlayStation 3, PlayStation 2, PlayStation Vita y plataformas PSP. Permite la utilización de características hardware específicas de dichas consolas como la ejecución en paralelo sobre su sistema de procesadores en Celda, un conjunto de procesadores RISC PowerPC de 64 bits a 3.2GHz de frecuencia y diseñado para una ejecución real en paralelo de los procesos. Se ofrece de forma gratuita a todos los desarrolladores licenciados por Sony como parte del SDK de PlayStation. Más de 90 juegos se han desarrollado con este motor como *Flower o Journey, AMY, Souls o Dark Souls* entre otros.

03.6. Otros Game engines

Tomando como referencia la arquitectura y herramientas empleadas para desarrollar ciertos juegos que se han convertido en referentes a lo largo de la historia de los videojuegos, se han popularizado y han evolucionado ciertas plataformas empleadas en su momento para crear dichos juegos específicos.

Por ejemplo ***El Motor de Juegos Half-Life*** se basa en el código empleado para la creación de *Half Life* 2 y sus secuelas (*Episode One, Episode Two, Team Fortress 2 o Portal)* y expone un motor de gran calidad a nivel de gráficos y herramientas que lo ponen a la altura de *Unreal Engine 4.*

Otro ejemplo se encuentra en el motor ***DICE Frostbite*** nacido en el año 2006 para el desarrollo del juego *Battlefield Bad Company y que* ha evolucionado para convertirse en la principal herramienta de desarrollo de la empresa *Electronics Arts (EA)*. Entre los títulos desarrollados con dicho motor se encuentran *Mass Effect, Battlefield, Need for Speed*, *Dragon Age o Command & Conquer.* Entre sus puntos fuertes se encuentra su herramienta *FrostEd* de creación de contenidos así como un potente motor en tiempo real y permite la generación de aplicaciones para PC, Xbox 360, Xbox One, PlayStation 3 y 4.

04. Arquitectura de los motores en tiempo de ejecución

Los grandes bloques en que se puede organizar un Game Engine se muestran en la siguiente imagen pudiéndose hacer uso de ellos según las necesidades

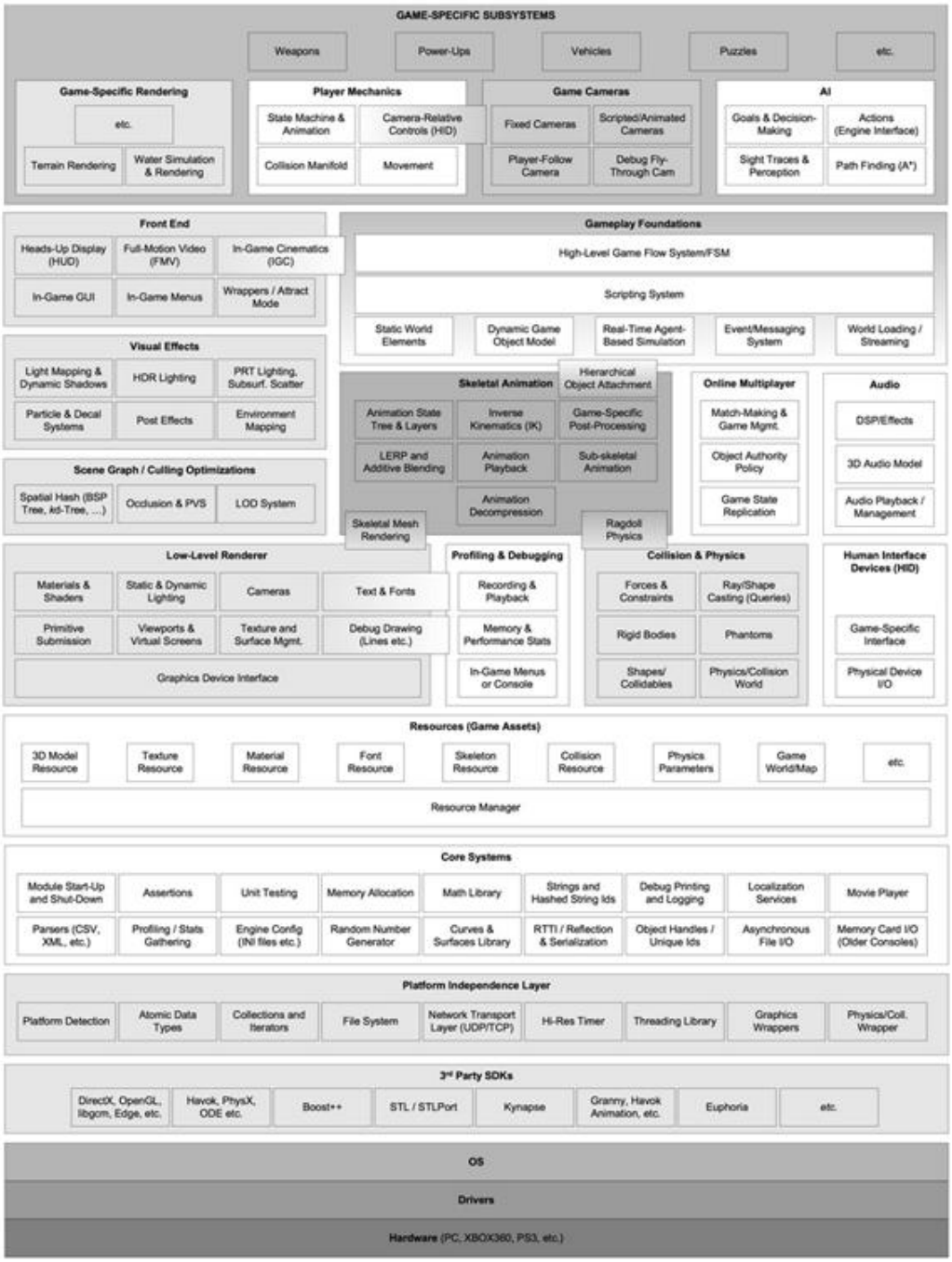

Figura 1: Arquitectura motor de juegos

04.1. Capas dependientes del dispositivo

Un gran sistema de este tipo está compuesto de capas donde se establecen dependencias entre distintas capas y algunas se pueden sustituir según conveniencia. Por ejemplo, en las capas superiores se encuentra la **capa Hardware Destino** y los **Drivers de dispositivos** que ofrece especializaciones particulares para cada Sistema Operativo final: plataformas de escritorio Microsoft Windows, Linux, y MacOS; mobile platforms, plataformas móviles como Apple iPhone, iPad, Android smart phones and tablets, Sony PlayStation Vita, Amazon's Kindle Fire¸etc. y consolas de juego como Microsoft's Xbox, Xbox 360 y Xbox One, Sony PlayStation, PlayStation 2, PlayStation 3 y PlayStation 4, y Nintendo's DS, GameCube, Wii y Wii U.

Por debajo estas capas se apoyan en capas comunes a todos ellos, por lo que las capas inferiores se pueden considerar agnósticas en cuanto a la plataforma.

04.2. Capas agnósticas

SDKs de terceros

La mayoría de los motores se apoyan en código de terceros (*Software Development Kits o SDK)* para dar soporte a diferentes áreas. El código se expone a través de librerías que exponen un API o interfaz de programación común. Por ejemplo algunas de las librerías comúnmente aceptadas son:

- La gestión de las estructuras de datos a través de librerías como *STL*, *Boost*, *Loki*, etc.
- Representación de gráficos confiando en librerías como *OpenGL o DirectX, libgem* (interface de bajo nivel para PlayStation 3), *Edge* (también para PlayStation 3), *Glide,* etc.
- Detección de colisiones y motores físicos para simulación de dinámicas de objetos usando librerías como *Havok, PhysX (*ofrecido por NVIDIA) *o Open Dynamics Engine (ODE)* como librería de código abierto.
- Animación de personajes siendo este un punto muy importancia para dotar a los juegos de realismo. Muchas empresas deciden desarrollar librerías propias, pero entre las disponibles se encuentran *Granny* que permite trabajar con los formatos de aplicaciones de diseño 3D como *Maya, 3D studio Max,* etc. manipulándolos en tiempo de diseño e importándolos en tiempo de ejecución, *Havok Animation, Edge* que aparte de funcionalidades de representación de gráficos ofrece API para animación, etc.

Motor de Renderizado

El motor de renderizado o de presentación es el componente más complejo de cualquier motor gráfico. Como en otras partes de la arquitectura, esta capa se divide habitualmente en subcapas para encargarse de aspectos específicos de la presentación y dependen a nivel más bajo de una capa dependiente del dispositivo para la representación gráfica como se puede ver en la Figura 1.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON Departamento de Innovación, Investigación y Universidad

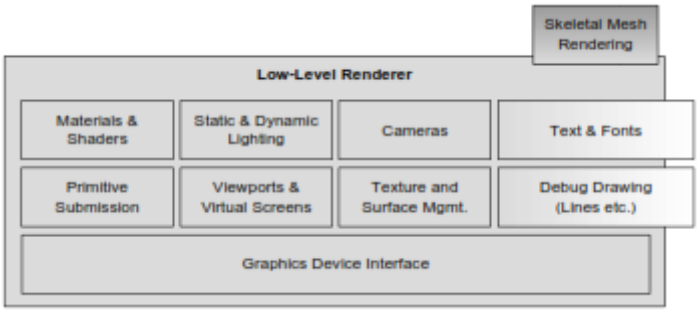

Figura 1: Arquitectura del motor de renderizado a bajo nivel

Capa de renderizado a bajo nivel

Esta capa se optimiza para dibujar de la forma más eficiente posible un conjunto de formas geométricas básicas de la forma más rica posible; sombras, texturas, luces, etc. se emplean aquí para dotar a dichas formas de aspecto deseado.

Librerías gráficas como DirectX, OpenGL se emplean para satisfacer estos requisitos. En muchas ocasiones, estas formas se agrupan en conjuntos como mayas con miles de triángulos, nubes de puntos, partículas, textos, etc. para optimizar su representación.

Esta capa se encarga también de trasladar la vista del mundo a través de una matriz de transformación a la vista de una o varias cámaras y a una proyección 3D tratando aspectos tales como el campo de vista y los planos de cortes cercanos y lejanos para evitar la representación de objetos no relevantes para la posición actual de las cámaras.

Los materiales y luces por ejemplo se gestionan por sistemas específicos de materiales y de iluminación dinámica y el resultado final es un flujo de procesamiento que se agrupa en primitivas gráficas que se envían a la tarjeta gráfica y esta se encarga de ir procesando; por ejemplo los materiales describen qué texturas se emplean en cada forma geométrica y qué shaders usar para dibujar cada pixel y vértice; el sistema de iluminación calcula como las luces del entorno afectan a la representación de los materiales; superficies metálicas, rugosas, etc. adquieren en esta fase su aspecto final.
Las tarjetas gráficas de última generación están optimizadas para procesar un gran número de primitivas gracias a sus procesadores especializados *GPUs.*

Capa de optimizaciones de escena

Si la capa de bajo nivel solo se preocupa de dibujar formas geométricas, más a alto nivel, se prioriza el objetivo de reducir el número de formas a representar con el consiguiente ahorro de tiempo de procesamiento que a la postre permitirá aumentar la calidad de lo que si se ha de representar.
Esta reducción se produce eliminando aquellas formas que no están visibles para la cámara en cada frame o aplicando otros algoritmos; un buen ejemplo es la arquitectura OGRE plug&play que permite añadir o eliminar algoritmos ya desarrollados o desarrollar algoritmos propios.

Efectos Visuales
Ciertos efectos requieren algoritmos específicos que fácilmente pueden ir sustituyéndose por nuevos que mejoren los resultados o incorporen más características. Sistemas de partículas (humo, fuego, agua, etc), sistemas de reproducción o calco para mostrar gran cantidad de elementos del mismo tipo como los impactos de bala, pisadas, etc. Anti aliasing para reducir los bordes recortados en las imágenes, efectos de luz como saturación, HDR (Hight Dynamic Range), etc. entrarían dentro de este grupo.

Capa frontal
Todos los motores gráficos componen una capa normalmente 2D para presentar información relevante al entorno, así como elementos gráficos que sirvan al usuario para interactuar con el sistema; en un juego, se emplea para mostrar los puntos conseguidos, los menús, las vidas restantes, etc.

05. Realidad Virtual y su integración con los motores de juegos

Realidad Virtual o VR por sus siglas en inglés han coexistido en el mundo de la tecnología desde los años 70 con el simulador *Flight Simulation*, pero la complejidad y elevado coste de las infraestructuras necesarias, la complejidad de cualquier desarrollo mínimo, y la falta de realidad del producto final, ha hecho que durante todos estos años el término haya aparecido en la escena tecnológica con la misma fuerza con la que ha desaparecido y se ha mantenido siempre latente, pero en reducidos círculos de desarrollo.

En los últimos años, la reducción de costes hardware, la simplificación en el desarrollo software mediante la reutilización de componentes y la evolución tecnológica a nivel de representación gráfica, procesamiento en GPUs, procesadores especializados cada vez más potentes, capacidad de almacenamiento, etc. junto con nuevas arquitecturas de desarrollo de software (principalmente a través de Game Engines) que se han adaptado fácilmente a este paradigma han conseguido popularizar el uso de esta tecnología.

Nuevos motores están apareciendo especializados en VR, mientras que los más populares se han sabido adaptar para dar soluciones a estos nuevos escenarios.

La forma de representar la realidad virtual se conseguía a través de la proyección de imágenes o múltiples pantallas que cubren la mayor parte posible de la visión del usuario como en los cines esféricos donde se sincroniza la proyección desde varios video proyectores para formar una imagen sobre una cúpula. Esta infraestructura es muy compleja y complicada de sincronizar y desde un punto de vista de procesamiento requiere que el mundo virtual se represente completamente a cada refresco de pantalla.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON Departamento de Innovación, Investigación y Universidad

Los *Cascos de Realidad Virtual* (*Head Mounted Display o HMD*) consisten en un dispositivo que el usuario se pone en la cabeza y recibe el mundo virtual a través de una pantalla situada frente a sus ojos. Uno de los retos que esto supone es que las aplicaciones deben conocer en todo momento la dirección en la que el usuario está mirando, señalando con las manos, etc. conocido con el nombre de *Seguimiento de Usuario o User Tracking* en inglés. Este conocimiento se obtiene de dispositivos hardware como Acelerómetros, giróscopos, magnetómetros, etc. Con este conocimiento, la aplicación puede optimizar sus esfuerzos en presentar en la pantalla solo el Mundo Visible por el usuario, manteniendo una representación en memoria (no visual) del resto del mundo) lo que permite un coste considerable en el tiempo de procesamiento de cada frame y permite construir mundos potencialmente infinitos.

Tecnologías emergentes hoy día permiten realizar este tracking de forma muy eficiente y existen ya en el mercado productos como Samsung Gear VR, Oculus Rift o el proyecto de Sony, denominado Morpheus, que cada uno con su enfoque y diferencias permiten desarrollar proyectos de Realidad Virtual con un alto grado de realismo y complejidad.

05.1. VR en los motores gráficos

Los distintos motores gráficos han ofrecido soporte a estos entornos tan específicos en sus distintas versiones, comprometiendo en mayor o menor medida la portabilidad de los desarrollos.

Unity ha soportado históricamente distintos entornos de VR, pero con ciertos problemas:

- Cada dispositivo de VR tenía un plugin específico, con lo que la misma aplicación tenía que adaptarse a cada entorno de ejecución.
- Distintos plugins para distintos dispositivos VR podían entrar en conflicto y el trabajo para soportar cada uno de estos dispositivos era considerable.
- Nuevas versiones de dichos plugins podían ser incompatibles hacia atrás con desarrolllos existentes.
- Desarrollar una aplicación para entornos VR y no VR a la vez era costoso.

La nueva versión de Unity (Unity 5.1) ofrece un API común para desarrollar en entornos VR dando soporte a ciertos dispositivos VR existentes en el mercado:

Esto simplifica el desarrollo multi dispositivo a unos pequeños cambios en la configuración, y si bien el API es sencillo, se irá adaptando a las necesidades de los desarrolladores según las necesidades que vayan surgiendo.

Dispositivo	Descripción
Stereo	Stereo 3D via D3D11 o OpenGL.
Split	Duplica la salida *side by side* (cámara izquierda y derecha).
Oculus	Dispositivos Oculus VR.
Morpheus	Dispositivo Sony Morpheus VR para Playstation 4

Figura 3: Tipos de dispositivos VR soportados por Unity de forma nativa

06. Aplicaciones basadas en motores gráficos y serious games

Como se ha visto, hasta ahora se ha explicado en detalle los motores gráficos con sus distintos componentes y cómo integran la parte de la Realidad Virtual. La idea es utilizar estas tecnologías cuyo origen primario es el campo del entretenimiento con otro tipo de propósitos como educativo, industrial, de simulación de procesos, etc.

A continuación, se van a explicar varios tipos de proyectos y ámbitos en los que se pueden realizar aplicaciones basadas en motores gráficos y las ventajas que presentan:

06.1. Aplicaciones basadas en motores gráficos para validar prototipos de diseño

En este campo la gran ventaja que presenta estas tecnologías es que permiten obtener un feedback rápido del producto que se está diseñando. Esto es muy interesante desde el punto de vista de las ingenierías ya que les permite a través de tecnologías de RV, cuando se pongan las gafas tener una visualización a escala real de los prototipos que han diseñado y por lo tanto tener un feedback rápido del diseño lo que permite tener iteraciones más cortas y eliminar errores antes de fabricar ninguna muestra.

Este uso también es interesante desde el punto de vista de los clientes ya que pueden ver lo que van a adquirir, se les puede mostrar los productos en sus instalaciones y, por lo tanto, entienden mejor lo que compran por lo que tienen gran aplicación enfocadas a marketing y ventas.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON Departamento de Innovación, Investigación y Universidad

06.2. Aplicaciones basadas en motores gráficos para simulaciones de elementos finitos

En este caso se utilizan las tecnologías para comprender mejor los resultados de una simulación ya que permiten tener visualizaciones a escala real y se tiene la posibilidad de animar estas simulaciones y ver los datos simulados además de explorar los resultados.

La gran ventaja es que permiten entender a través de la tecnología fenómenos complejos y presentárselos a personas no expertas ya que la Realidad Virtual está ayudando en el proceso de visualización de los datos.

06.3. Serious games para entrenamiento o formación

Este último ejemplo es el más extendido y ha tenido un largo recorrido en sectores como el militar e industrial gracias a que los simuladores permiten recrear situaciones próximas a la realidad de una forma mucho más segura que el entrenamiento en un entorno real.

En este sentido muchas de las situaciones en las que el trabajador se encontrará no pueden ser practicadas en un entorno real por conllevar un riesgo y por lo tanto, el uso de simuladores permite al trabajador entrenarse repetidamente en condiciones que podrían ser peligrosas adquiriendo unas habilidades, de forma que cuando el trabajador se encuentre con esa situación en la vida real sepa reaccionar adecuadamente ya que la ha practicado multitud de veces en el simulador.

Por otro lado el uso de simuladores evita que trabajadores inexpertos o no suficientemente preparados trabajen con maquinaria potencialmente peligrosa sin el suficiente número de horas de práctica necesaria para que se realice con seguridad. Esto normalmente viene motivado por el alto coste que significa que una máquina se dedique a la formación de un trabajador en vez de que la máquina está produciendo ya que se intenta optimizar el rendimiento de la maquinaria al máximo.

Dadas las razones antes enumeradas, la reducción de los costes de la realidad Virtual, añadido al hecho de que van a ser productos de gran consumo, posibilitan el desarrollo de simuladores con motores gráficos 3D comerciales que pueden correr en estaciones de trabajo de gama media y que no requieren infraestructuras específicas.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON Departamento de Innovación, Investigación y Universidad

07. Conclusiones

A lo largo de este documento se ha presentado un estado del arte presentando la arquitectura y componentes principales de los motores de juegos (game engines) y se han presentado distintos motores gráficos: Cry Engine, Unity, Unreal Engine, Microsoft XNA, etc.

Posteriormente, se ha explicado la integración que están experimentando los motores gráficos para integrarlos con las soluciones de Realidad Virtual como Oculus Rift, Gear y Morpheus, particularizándolo para el motor Unity.

Por último se ha querido presentar varios tipos de proyectos y cómo estas tecnologías de RV pueden aplicar a los mismos en entornos de diseño de prototipos, simulación de elementos finitos y serious games para entrenamiento o formación.

GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
GOBIERNO DE ARAGON
Departamento de Innovación, Investigación y Universidad

08. Referencias

[1] Game Engine Book, http://www.gameenginebook.com/figures.html

[2] Jason Gregory, *Game Engine Architecture*, 2009.

[3] Alan B. Craig, William R Sherman, and Jefrey D. Will, “Developing Virtual Reality Applications” 2 (2013).

[4] Motor gráfico Untiy, https://unity3d.com/es

[5] Motor gráfico Unreal, https://www.unrealengine.com/blog

[6] Oculus Rift, https://www.oculus.com/en-us/

[7] Proyecto Morpheus, https://www.playstation.com/en-us/explore/project-morpheus/